

Deep Neural Networks to Enable Real-time Multimessenger Astrophysics

Daniel George^{1,2} and E. A. Huerta²

¹*Department of Astronomy, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801, USA*

²*NCSA, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801, USA*

(Dated: January 5, 2017)

We introduce a new method for time-domain signal processing, based on deep learning neural networks, which has the potential to revolutionize data analysis in science. To demonstrate how this enables real-time multimessenger astrophysics, we designed two deep convolutional neural networks that can analyze time-series data from observatories including advanced LIGO. The first neural network recognizes the presence of gravitational waves from binary black hole mergers, and the second one estimates the mass of each black hole, given weak signals hidden in extremely noisy time-series inputs. We highlight the advantages offered by this novel method, which outperforms matched-filtering or conventional machine learning techniques, and propose strategies to extend our implementation for simultaneously targeting different classes of gravitational wave sources while ignoring anomalous noise transients. Our results strongly indicate that deep neural networks are highly efficient and versatile tools for directly processing any raw noisy data streams. We also pioneer a new paradigm to accelerate scientific discovery by combining high-performance simulations on traditional supercomputers and artificial intelligence algorithms that exploit innovative hardware architectures such as deep-learning-optimized GPUs. This unique approach immediately provides a natural framework to unify multi-spectrum observations in real-time thus enabling coincident detection campaigns of gravitational waves sources and their electromagnetic counterparts.

I. INTRODUCTION

Gravitational wave (GW) science has reshaped the landscape of observational astronomy and theoretical astrophysics. During the first observing run of the advanced Laser Interferometric Gravitational wave Observatory (aLIGO) detectors, we peered into the most powerful events in the cosmos driven by extreme gravitational interactions [1–3]. The consequent direct detections of GWs from the mergers of binary black hole (BBH) systems provided the first glimpse of the nature of gravity in the strong-field regime. It offered the earliest observational evidence for the formation and merger of BBHs within a Hubble time, confirmed the existence of massive stellar-mass BHs, and provided the first direct measurement of the angular momentum of BHs [2–12].

These studies relied on accurate massively-parallel numerical relativity (NR) simulations of Einstein’s field equations, which served a dual purpose: (i) they conclusively showed that Einstein’s general relativity can describe with outstanding accuracy the true nature of gravity in the most extreme astrophysical environments and (ii) having thus established its status, comparison with signal templates generated or calibrated with NR is extensively used to infer astrophysical properties of the detected GW sources [13, 14].

After this flurry of breakthroughs, it is apparent that the detection of GWs will become a common occurrence as the aLIGO detectors gradually attain design sensitivity in the following years. Furthermore, a worldwide network of kilometer-scale GW detectors in the US, Europe and Asia will considerably increase the science reach of GW astrophysics [15–18].

In the near future, the discovery of vanilla BBH systems will be important for stringent tests of general rel-

ativity and to get better insights into stellar evolution processes. However, the emphasis of multimessenger astrophysics will be on *extraordinary* events, such as the mergers of binary neutron stars (BNSs), NSBH, intermediate mass black holes (IMBHs) with masses between $100M_{\odot} - 500M_{\odot}$, IMBHs and stellar mass BHs or NSs, as well as eccentric binary coalescences, core-collapse supernovae, and other exotic unexpected events. Joint discovery campaigns with powerful optical detectors such as the Dark Energy Survey (DES) [19], the Large Synoptic Survey Telescope (LSST) [20], Euclid [21], and WFIRST [22] will enable the coincident detection of GW events which are expected to generate electromagnetic (EM) counterparts.

Bringing multimessenger astrophysics into fruition requires the combination of multi-spectrum observations with very low latency. Detecting a GW candidate in real-time and establishing whether its parameters correspond to likely progenitors of EM counterparts [23–29] is essential to enable rapid broadband EM and astroparticle searches [30, 31]. To this effect, aLIGO’s online GW detection pipelines are allocated substantial computational resources, ranging over hundreds of compute nodes at LIGO Data Centers, so that they can keep up with the huge influx of data from intensive GW detection campaigns.

In this article, we introduce a new kind of approach which we discovered for processing highly noisy time-series data using deep neural networks (DNNs), that differs significantly from currently used matched-filtering and machine learning techniques. When applied for the detection and characterization of GW sources, this provides an ideal framework to expand the depth of current aLIGO flagship algorithms [32–40], allowing simultaneous real-time searches for compact binary coalescence

and GW bursts under a single unified pipeline. As opposed to existing aLIGO search pipelines requiring thousands of dedicated CPU cores to operate, a DNN pipeline can potentially be run on a single CPU core and very intensive searches over a broader range of signals can be easily carried out with a few dedicated GPUs. Furthermore, since this approach can be seamlessly applied to any type of raw data, DNNs can also be used to search for transient events in data gathered by other astronomical facilities using on-site embedded GPU hardware, thus paving a natural path to realizing multimessenger astrophysics.

The application of DNNs in GW astrophysics, astronomy, and astro-particle physics has the potential to accelerate scientific research and unlock new opportunities by enhancing the way we use existing High Performance Computing (HPC) resources while allowing us to exploit emerging hardware architectures such as deep-learning-optimized Graphics Processing Units (GPUs) [41] and Field-Programmable Gate Arrays (FPGAs) [42]. Working in tandem with computer scientists and industries to develop Artificial Intelligence (AI) tools that extend our prototype, and further exploring applications of deep learning for multimessenger astrophysics and fundamental sciences, may provide the means to effectively consolidate different windows of observation into the Universe.

Executive Summary

Extracting GW signals from the detectors’ data stream requires the use of algorithms that can distinguish signals whose amplitude is much smaller than the noise background. The detector output has two classes of noise in the calibrated strain data: (i) the main component is Gaussian noise, which originates from seismic, thermal, and quantum processes and (ii) non-Gaussian noise from instrumental and environmental factors [32]. Prior to carrying out GW searches, data quality investigations are implemented to remove segments of poor quality that are not suitable for data analysis.

aLIGO’s flagship matched-filtering searches are based on the construction of a template bank of waveforms that describe quasi-circular compact binaries whose components are spin-aligned [32]. There are ongoing efforts to extend the parameter-space which can be probed with these algorithms, particularly in the context of spin-precessing BBHs [43], and searches that include higher-order waveform multipoles [44]. However, there remain substantial difficulties in achieving this work and implementing these algorithms in future GW searches. Another limitation of available aLIGO pipelines is the use of a single noise Power Spectral Density (PSD), averaged over all the detectors for the length of the search, which was necessary to computationally optimize matched-filtering techniques [32].

Our deep learning algorithm aims to overcome these limitations while offering additional advantages. In this

initial study, we carried out a systematic assessment of DNN-based algorithms over a restricted parameter-space of signals and our results confirm that DNNs are ideal tools for future GW analysis. Since all the intensive computation is diverted to the training stage of DNNs, much larger template banks may be used and the actual analysis can be carried out in real-time with minimal resources. The intelligent nature of our algorithm allows automated learning of persistent and transient characteristics of noises inherent to the detectors while incorporating data quality information. Most importantly, it enables creating a single pipeline for performing all tasks — identifying the presence or absence of GW signals, classifying noise transients, and reconstructing the astrophysical parameters of detected GW sources.

The size of the template bank used for training is determined by the accuracy with which one desires to recover the parameters of GW sources. Since the DNN algorithm can learn to interpolate, it is capable of generating new waveforms to densely populate the parameter-space under consideration in a similar manner to Gaussian Process Regression (GPR) ¹.

In addition, the performance of DNNs monotonically improves when trained with massive quantities of data while the rate of evaluation remains constant. Training a network with a large template bank of gravitational waveforms may take several days on powerful modern hardware, but once the training is complete — no matter how many parameters are taken into account, i.e., spin-precession, eccentricity, higher-order modes, etc. — the pipeline would be capable of detecting the presence of a GW signal in noisy data within milliseconds. Moreover, it is possible to *re-train* the entire DNN-pipeline within minutes to use slightly different PSDs and inputs from real-time data quality investigations.

To summarize, our novel DNN algorithm departs significantly from existing GW pipelines and provides a whole new framework for GW detection that will enhance and optimize the use of high-throughput and HPC resources for scientific data analysis. Furthermore, since we can target a wide variety of GW sources using a single DNN pipeline, our algorithm is ideal for unified GW searches ranging from GW bursts, to compact binary mergers, to supernovae and other exotic events. In order to train DNNs with waveforms that faithfully describe the gravitational and EM signatures of various sources, we will continue to rely on NR simulations. The future of multimessenger astronomy depends critically on the optimal use of supercomputing resources and sophisticated hardware technologies in combination with AI algorithms. The methodology we present here constitutes the first pivotal step in that direction.

¹ GPR [45–47] is a statistical tool that can serve as a probabilistic interpolation algorithm providing information about the training set of NR simulations needed to accurately describe a given parameter-space and generates interpolated waveforms that match NR counterparts above any given accuracy.

This article is organized as follows: Section II provides an introduction to deep learning and DNNs. In Section III, we outline a proof of concept of our DNN-based GW detection pipeline in the context of quasi-circular, non-spinning BBH mergers. Section IV recounts the procedure followed to construct datasets and describes our strategy to design and train DNNs. We report the results of this analysis in Section V. In Section VI, we explore the impact of applying DNNs for multimessenger astrophysics in the context of astronomical instruments, advanced cyberinfrastructure facilities for data analysis, and new hardware architectures tailored for deep learning. We summarize our findings and discuss avenues to expand this new paradigm in Section VII.

II. DEEP NEURAL NETWORKS

Although deep learning with neural networks is rapidly becoming a ubiquitous technology in industrial applications, it remains a relatively new and unfamiliar topic in the fundamental sciences. Therefore, in this section, we provide a brief overview of the main concepts of deep learning, including machine learning, artificial neural networks, and convolutional neural networks in the context of signal processing.

Machine Learning

The vast majority of algorithms are designed with a specific task in mind. They require extensive modifications before they can be re-used for any other task. The term machine learning refers to a special class of algorithms that can *learn* from examples to solve new problems without being explicitly re-programmed. This enables cross-domain applications of the same algorithm by simply training it with different data. More importantly, some of these algorithms are able to tackle problems which humans can solve intuitively but find difficult to explain using well-defined rules, hence they are often called “artificial intelligence” [48].

The two main categories of machine learning are supervised and unsupervised learning. In supervised learning, the algorithm learns from data that is correctly labeled, while unsupervised learning algorithms have to make sense of unstructured and unlabeled data [49]. We will be focusing on an application of supervised learning in this work, where we use labeled data obtained from physics simulations to train an algorithm to detect signals embedded in noise and also estimate multiple parameters of the source.

Although traditional machine learning algorithms have been successful in several applications, they are limited in their ability to deal directly with raw data. Often the data has to be simplified manually into a representation suitable for each problem. Determining the right representation is extremely difficult and time-consuming,

often requiring decades of effort even for domain experts. This severely limits the usage of these algorithms [48].

Representation learning is a subset of machine learning which aims to resolve this issue by creating algorithms that can learn by themselves to find useful representations of the raw data and extract relevant features from it automatically for each problem [50]. Here, we are focusing on a special type of representation learning called deep learning.

Deep Learning

Deep learning combines a computational architecture containing long interconnected layers of “artificial neural networks” with powerful learning algorithms [51]. These deep artificial neural networks are able to capture complex non-linear relationships in the data using hierarchical internal representations, all of which are learned automatically during the training stage. The deepest layers are able to learn highly abstract concepts based on the simpler outputs of the previous layers. This ability has enabled deep learning to solve a wide range of problems that were previously thought to require human-level intelligence [49].

Various factors including the exponential growth of computational resources (especially GPUs), availability of massive amounts of data, and the development of new algorithmic techniques and software have recently contributed to make deep learning very successful in commercial applications, thus revolutionizing multiple industries today. The state-of-the-art algorithms for image processing, speech recognition, natural language understanding are all based on deep learning. DNNs power many of the technologies routinely used by us including search engines (Google, Bing), voice recognition on smartphones, personal assistants (Siri, Cortana, Google assistant), mobile keyboards (SwiftKey), real-time face detection on cameras, face recognition (Facebook), language translation (Google Translate), text-to-speech synthesis [52], recommendations on Amazon, and automatic captioning on YouTube, to name a few [53].

Most notably, deep learning was used in combination with reinforcement learning [54] to build a program called AlphaGo [55] which defeated one of the world’s best players, in 2016, at the highly complex game of Go. Yet another recent success was at lip reading, where an algorithm has surpassed the best humans by a large margin of accuracy [56]. Deep learning is also the key ingredient in self-driving vehicles that are being deployed now by Tesla, Uber, and Google.

Artificial Neural Networks

Artificial neural networks (ANN), the building blocks of DNNs, are biologically-inspired computational models that have the capability to learn from observational

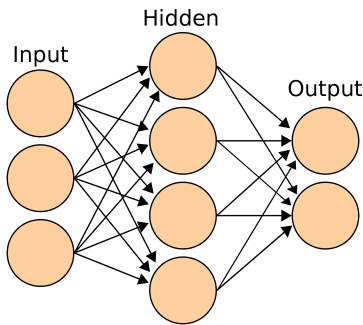


FIG. 1. An Artificial Neural Network (ANN) or multilayer perceptron with one hidden layer is depicted [62]. The circles represent neurons and arrows represent connections between neurons. Note that each neuron has only a single output which branches out to connect with neurons in the next layer.

data [57]. The fundamental units of neural networks are artificial neurons (loosely modeled after real neurons [58]), which are based on perceptrons introduced by Rosenblatt in 1957 [59]. A perceptron takes a vector of inputs (\vec{x}) and computes a weighted output with an offset known as bias. This can be modeled by the equation $f(\vec{x}) = \vec{w} \cdot \vec{x} + b$, where the weights (\vec{w}) and bias (b) are learned through training.

Minsky and Papert showed in their 1969 book *Perceptrons* [60] that a single perceptron has many limitations. Unfortunately, this led to a decline in the popularity of all neural networks in the following decades [49]. However, it was later found that these limitations can be overcome by using multiple layers of inter-connected perceptrons to create ANNs. The universality theorem [61] proves that ANNs with just three layers (one hidden layer) can model any function up to any desired level of accuracy.

These multilayer perceptrons are also known as feed-forward neural networks because information is propagated forward from the input layer to the output layer without internal cycles (i.e no feedback loops) [48]. While potentially more powerful cyclic architectures can be constructed, such as Recurrent Neural Networks (RNNs), we will be focusing mainly on feed-forward neural networks in this article.

An ANN usually has an input layer, one or more hidden layers, and an output layer (shown in Figure 1). A non-linear “activation” function is applied to the output of each of the hidden layers. Without this non-linearity, using multiple layers would become redundant, as the network will only be able to express linear combinations of the input. The most commonly used non-linear activation functions are the logistic sigmoid, hyperbolic tan, and the rectified linear unit (also called ReLU or ramp). It has been empirically observed that the ramp produces the best results for most applications [63]. This function is mathematically expressed as $\max(0, x)$.

The key ingredient that makes ANNs useful is the learning algorithm. Almost all neural networks used to-

day are trained with variants of the back-propagation algorithm based on the steepest descent method [49]. The idea is to propagate errors backward from the output layer to the input layer after each evaluation of a neural network, in order to adjust the weights of each neuron so that the overall error is reduced in a supervised learning problem [64]. The weights of an ANN are usually initialized randomly to small values and then back-propagation is performed over multiple rounds, known as epochs, until the errors are minimized. Stochastic gradient descent with mini-batches [65] has been the traditional method used for back-propagation. This technique uses an estimate of the gradient of the error over subsets of the training data in each iteration to change the weights of the ANN. The magnitude of these changes is determined by the “learning rate”. New methods with variable learning rates such as ADAM (Adaptive Momentum Estimation) are becoming more popular and have been shown empirically to achieve better results more quickly [66].

Convolutional Neural Networks

A convolutional neural network (CNN) is a type of feed-forward neural network largely responsible for the current wave of success enjoyed by DNNs. First developed by Fukushima for his Neocognitron [67], they were successfully combined with back-propagation by LeCun [68] in the 1980s, for developing a highly accurate algorithm for recognizing handwritten digits. These CNNs were used by banks for processing over 10% of all checks in the US during the early 2000s [48]. However, their full potential was not discovered for another couple of decades mainly due to hardware constraints and lack of large amounts of high quality data at the time. The exceptional performance of Alex Krizhevsky’s entry based on CNNs, which won the ImageNet competition by a huge margin in 2012 [69], has sparked the current interest in these networks especially in the field of computer vision. Until now, CNNs have been most effective for image and video processing. They have been shown to approach or even surpass human-level accuracy at a variety of constrained tasks such as hand-writing recognition, identifying objects in photos, tracking movements in videos etc. [51].

The introduction of a “convolution layer”, containing a set of neurons that share their weights, is the critical component of these networks. Multiple convolution layers are commonly found in DNNs, with each having a separate set of shared weights that are learned during training. The name comes from the fact that an output equivalent to a convolution (or sometimes cross-correlation [48]) operation is computed with a kernel of fixed size. A convolutional layer can also be viewed as a layer of identical neurons that each “look” at small overlapping sections of the input (defined as the receptive field). In fact, they were inspired by studies of the visual cortex in mammals [48].

The main advantage of using these layers is the ability to reduce computational costs by having shared weights and small kernels, thus allowing deeper networks and faster training and evaluation speeds. Because of the replicated structure, CNNs are also able to automatically deal with spatially translated as well as (with a few modifications [51]) rotated and scaled signals.

In practice, multiple modules each consisting of a sequence of convolution and pooling (sub-sampling) layers, followed by a non-linearity, are commonly used. The most popular form of pooling is max-pooling, where only the maximum value within a fixed kernel size is chosen. Addition of the pooling layers reduces computational costs while also making the networks resilient to small-scale noise, thus enhancing their ability to handle new inputs [51].

Signal Processing with Deep Neural Networks

Conventional methods of digital signal processing such as matched-filtering [70] (cross-correlation or convolution against a set of templates) in time-domain or frequency-space are limited in their ability to scale to a large parameter-space of signal templates, while being too computationally intensive for real-time analysis. Signal processing using machine learning is an emerging field of research [71–75]. These traditional machine learning techniques, including shallow ANNs, require “hand-crafted” features extracted from the data as inputs rather than the raw noisy data itself. DNNs, on the other hand, are capable of extracting these features automatically.

RNNs [49] are ideal for processing temporal data as they can take inputs of variable lengths. They have been remarkably successful at voice recognition problems [76]. However, they are harder to train [77] and the technology is still in the early stages, therefore we hope to revisit them in the future. We focus solely on CNNs in this study owing to their efficient implementation on modern hardware and their ability to automatically recognize translated inputs. Although CNNs have been used previously for classification of events [78–81], there has only been a few attempts at signal processing using CNNs with raw time-series data for parameter estimation [82, 83].

Signal processing is often done in the frequency domain. Fast Fourier Transform (FFT) algorithms are used for this conversion. Most published literature on time-series processing with CNNs have used spectrograms as inputs [84]. These images can easily be classified by many standard CNNs designed for object recognition (e.g. AlexNet [69], GoogLeNet [85], VGG [86], ResNet [87]). Doing so is advantageous for two reasons: (i) these architectures have been shown to work and (ii) partially trained weights are available for them, which can significantly speed up the training process while also providing higher accuracy even for small datasets.

However, when experimenting with spectrogram images as inputs, we found that a large amount of informa-

tion is absent and this severely limits the detectability of weak signals, having Signal-to-Noise Ratio (SNR) less than one ², that are not visible in spectrograms as shown in Figure 2.

Theoretically, all the information about the signal is encoded within the time-series. We discovered that by directly feeding raw time-series data as inputs to certain types of CNNs, one can obtain much higher accuracies and faster analysis rates. This approach is called automatic feature learning from data and allows the algorithm to develop more optimal strategies of signal processing rather than making judgments based on hand-extracted information such as periodograms or spectrograms.

In this work, we demonstrate that DNNs can be used for both signal detection and parameter estimation from noisy time-series data, when trained with templates of the expected signals, and that they outperform traditional machine learning algorithms and reach accuracies competitive to matched-filtering methods. To the best of our knowledge, this is the first scientific application of DNNs for the estimation of multiple parameters from raw time-series data using an end-to-end learning approach.

We also show that our algorithm is far more computationally efficient compared to matched-filtering. Instead of repeatedly performing overlap computations against each template of known signals, the deep CNN builds a deep *non-linear* hierarchical structure of nested convolutions that determines the parameters of the best matching template in a single evaluation thus allowing the analysis to be performed faster by several orders of magnitude. The evaluation rate of any neural network is independent of the size of the template bank used for training. Moreover, the DNN acts as an effective compression mechanism by learning patterns and encoding all the relevant information needed to distinguish different signals in their weights, which is several orders of magnitude smaller than the size of the original template bank. Therefore, the DNNs automatically perform an internal optimization of the search algorithm and can also interpolate to unseen parameter values. Multi-task learning [88, 89] can further minimize resources by allowing a single DNN to perform detection, classification, and parameter estimation. These properties enable scaling real-time analysis to a larger parameter-space of signals.

III. METHODOLOGY

Our goal is to apply this approach for GW analysis by demonstrating that algorithms based on DNNs are

² We are using the standard definition of SNR, which is equivalent to the ratio of the peak power of the signal to the root-mean-squared power of the noise; this is different from the formula used in the context of matched-filtering algorithms.

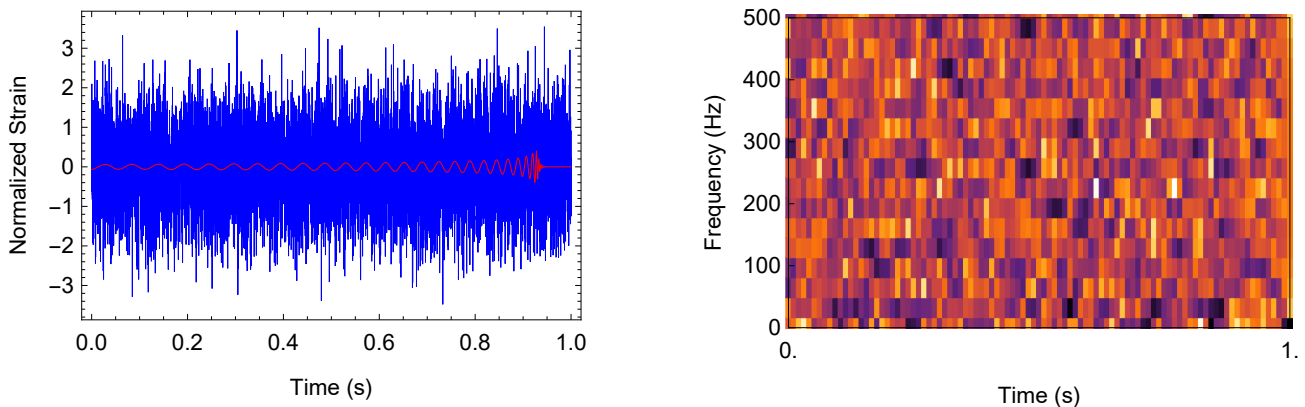


FIG. 2. Left panel: The blue time-series is a sample input to our DNN algorithm. It contains a BBH GW signal (red) which was whitened with aLIGO’s PSD design sensitivity (shown in Figure 3) and superimposed in noisy data with $\text{SNR} = 0.5$. Right panel: The corresponding spectrogram showing that the GW signal on the left is not visible and thus cannot be detected by any algorithm trained for image recognition. Nevertheless, our DNN detects the presence of this signal from the time-series input and reconstructs the source’s parameters with excellent accuracy.

ideal for detecting GW signals embedded in noisy time-series strain data from detectors, and classifying them into known categories while also being able to estimate the parameters of the source of the signal. Furthermore, we aim to show that, once trained, these DNNs are lightweight and fast enough to permit real-time analysis using minimal computational resources.

Approach

To provide a proof of concept, we avoid excessive complexity in this introductory article by focusing solely on BBH mergers. Nevertheless, we believe that it is straightforward to extend this method to signals produced by any type of event; we have outlined the necessary steps to accomplish this in Section VI.

We chose to divide the problem into two independent parts, each assigned to different DNNs. The first network, henceforth known as the “classifier”, will detect the presence of a signal in the input and provide a confidence level for the detection. The second network, which we call the “predictor”, will estimate the parameters of the source of the signal. The predictor is used if and only if the classifier identifies a signal with a high probability.

We have partitioned the system in this manner so that, in the future, different classes of events including BNS inspirals, supernovae, cosmic strings, etc., may be added to the same classifier and separate predictors can be made for each type of event. Moreover, categories for different types of anomalous sources of noise, like glitches [38, 75], can also be added to the classifier. Given sufficient examples of labeled templates of events and glitches, this can immediately be implemented within our framework.

The classes chosen for now are “True” or “False” depending on whether or not a signal from a BBH merger

is present in the input. Note that since we are training the network to recognize only BBH mergers, GWs from other events or glitches may be classified incorrectly by our model.

Once the classifier confirms that the signal originated from a BBH merger, the predictor is called upon to estimate the parameters of the source. Parameter estimation is also referred to as a regression task in machine learning literature [48]. This is a much harder problem than classification since the parameter-space is continuous as opposed to a finite discrete set of classes. Fine-scale features of the signal have to be measured over longer time periods in order to estimate the parameters whereas this is not necessary for simply detecting the presence of a signal. We will demonstrate that DNNs are excellent for all these tasks.

Assumptions

Ideally, the inputs to our DNNs will be the unprocessed time-series of strains measured by the GW detectors. Although we are using simulated white noise for this analysis, in order to emulate the form of the signals received by the detectors, we have weighted the gravitational waveforms used for the training set with the target design sensitivity for aLIGO — see Figure 3.

For this preliminary study, we are restricting our parameter-space to a smaller subset. The detectors’ strain is given by [91]: $h(t) = h_+(t)F_+ + h_\times(t)F_\times$, where $F_{+,\times}$ represent the antenna pattern of the detectors. We assume optimally oriented systems, which satisfy $F_+ = 1$, $F_\times = 0$.

There are three components for the spin of each of the compact objects. This accounts for a total of six parameters. Another free parameter is the orbital eccentricity of

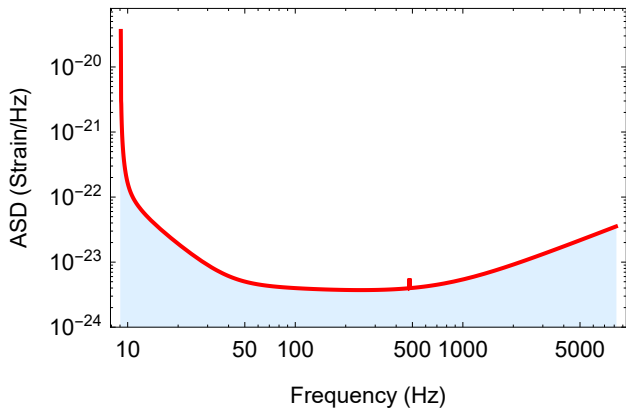


FIG. 3. Throughout this analysis, we have used the Zero Detuned High Power sensitivity configuration for aLIGO [90] shown above.

the binary. At present LIGO data analysis pipelines only use templates for systems with aligned spins and zero eccentricity. There is an ongoing effort by many groups (including the authors) to produce catalogs of numerical relativity simulations of BBH systems with non-aligned spins and significant eccentricities, which will be used in the future to train our DNNs. For now, we have assumed that the individual spins and eccentricities are zero thus reducing our parameter-space to two dimensions, namely, the individual masses of the black-holes, which we have restricted to lie between $5M_{\odot}$ and $75M_{\odot}$. Furthermore, we have constrained the inputs to have a duration of one second and a sampling rate of 8192 Hz throughout this analysis, since this is sufficient for the events that we are considering³.

IV. PROCEDURE

In this section, we describe in detail the steps followed to collect the datasets for training, validation, and testing, to design the DNNs, and to train them.

Obtaining Data

Supervised deep learning algorithms are far more effective when trained with massive amounts of data. Obtaining high quality training data has been a difficult and cumbersome task in most applications of DNNs such as object recognition in images, speech and text processing, etc. Fortunately, we do not face this issue since we can

³ Note that our method can be extended to any number of parameters with minimal modifications to the underlying code. The only difference would be changing the training set, adding more output neurons for each new parameter, and perhaps increasing the size of the existing layers.

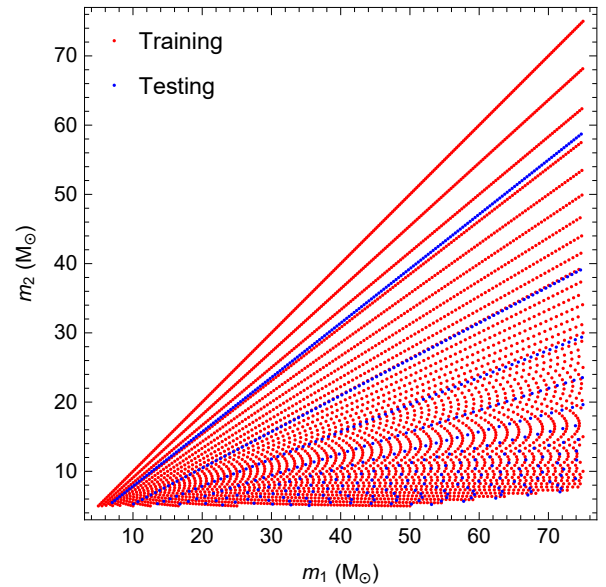


FIG. 4. The figure shows the distribution of component masses of BBHs chosen for our training and testing datasets.

take advantage of scientific simulations to produce the necessary data.

Over the last decade, multiple groups have successfully developed advanced techniques to use supercomputers to generate catalogs of highly accurate 3-dimensional numerical relativity simulations of merging BHs [92, 93]. This is further augmented by other methods such as post-Newtonian expansions for the inspiral [94] and BH perturbation theory [95–97] for the ringdown stages. Therefore, one can generate sufficient numbers of gravitational waveforms covering a wide range of parameters for training our DNNs.

For the analysis at hand, we use effective-one-body (EOB) waveforms, which are calibrated with NR simulations, that describe quasi-circular, non-spinning BBHs [98]. We extracted one-second windows around the peak of each waveform for our analysis.

The first step in a supervised learning problem is to split the data into separate sets for training and testing. We ensured that the mass-ratios of our binaries were values between 1 and 10 in steps of 0.1 for training, and randomly sampled mass-ratios lying in the same range for testing. By not having overlapping values in the training and testing sets, one can ensure that the network is not overfitting, i.e., memorizing only the inputs shown to it without learning to generalize to new inputs. The distribution of component masses is shown in Figure 4.

Subsequently, we shifted the location of the peak of each signal randomly within the last quarter second to make the predictor more robust with respect to time translations. This step may be avoided in the future since the fast evaluation speed of our algorithm makes it possible to use a sliding-window technique, so that the peak can be aligned within any desired region. Next, we

superimposed Gaussian white noise on top of the signals over multiple iterations, thus amplifying the size of the datasets. The power of the noise was adjusted according to the desired SNR. Since the overall amplitude of the signal only depends on the distance to the source for the parameters that we are considering, we ensured that our inputs are standardized to have zero mean and unit variance as this makes the training process easier [99].

The final training set contained approximately 100,000 time-series vectors produced from about 4000 templates of BBH mergers of different masses after adding white noise and translating in time. It is also a standard practice to use a validation set (also called a development set) to monitor the performance on unseen data during training in order to prevent overfitting. The validation set and testing sets were taken from the same distribution of component masses (following Andrew Ng’s suggestion [100]) albeit with different noise. These sets contained about 25,000 elements each.

Designing Neural Networks

We used the same neural network architecture for both the classifier and predictor, which demonstrates the versatility of our method. The only difference was the addition of a softmax layer [54] to the classifier to obtain probability estimates of the outputs. Our strategy was to first train the predictor on various BBH merger waveforms and then use this pre-trained network, with the added softmax layer, to train the classifier on datasets appended with 50% random noise. This significantly reduced the training time required for the classifier while also improving its accuracy at low SNR.

We experimented with a wide variety of designs for our DNN. Our initial attempt was to take some of the well-known architectures for image recognition, such as GoogLeNet, VGG, AlexNet, ResNet, and prune the layers to take one dimensional inputs. However, we discovered that the level of complexity needed for images was hardly required for our problem. It only served to slow down the training and evaluation speeds with negligible improvements in accuracy, if any. Instead, we decided to design simple DNN architectures from scratch. This approach yielded the best accuracy as well as the fastest training and evaluation rates.

Overall, we tested the accuracy of about 80 different configurations of DNNs ranging from 1 to 4 convolutional layers and 1 to 3 fully connected layers (also called linear layers). We discovered that 3 convolutional layers followed by 2 fully connected layers yielded the best results for the small parameter-space that we are considering. It is expected that as we increase the dimensions of the parameters and types of signals, we may need deeper and wider networks (note that many modern DNNs are several hundred layers deep).

Max-pooling layers were used after each convolution layer, to shrink the size of the output, followed by a ramp

	Input	vector (size: 8192)
1	Reshape Layer	tensor (size: $1 \times 1 \times 8192$)
2	Convolution Layer	tensor (size: $16 \times 1 \times 8177$)
3	Pooling Layer	tensor (size: $16 \times 1 \times 2045$)
4	Ramp	tensor (size: $16 \times 1 \times 2045$)
5	Convolution Layer	tensor (size: $32 \times 1 \times 2017$)
6	Pooling Layer	tensor (size: $32 \times 1 \times 505$)
7	Ramp	tensor (size: $32 \times 1 \times 505$)
8	Convolution Layer	tensor (size: $64 \times 1 \times 477$)
9	Pooling Layer	tensor (size: $64 \times 1 \times 120$)
10	Ramp	tensor (size: $64 \times 1 \times 120$)
11	Flatten Layer	vector (size: 7680)
12	Linear Layer	vector (size: 64)
13	Ramp	vector (size: 64)
14	Linear Layer	vector (size: 2)
15	Softmax Layer	vector (size: 2)
	Output	vector (size: 2)

FIG. 5. This is the layout of the DNN that we designed for classification. For prediction we simply replaced the 15th layer with a ramp (ReLU) function. The input is the time-series sampled at 8192Hz and the output is the probability of each class or the value of each parameter. Note that the size of layer 14 can be increased in order to add more categories for classification or more parameters for prediction.

(ReLU). We also tried adding a few recent developments such as batch normalization [101] and dropout [102] layers. However, we did not use them in our final design as they provided minimal improvements for the simple problem we are considering.

Many of the layers such as convolutional layers, pooling layers, and fully connected layers have parameters, commonly known as hyperparameters, that needs to be tuned manually. Determining ways to find optimal hyperparameters for neural networks is an area of active research [49]. At present, most groups use randomized trial and error based methods for determining these hyperparameters [57]. Another strategy that has recently gained popularity is Bayesian optimization [103], which is a probabilistic method to find values that optimize any noisy black-box function having a smooth distribution. We experimented with both techniques for choosing the following hyperparameters.

Depth is a hyperparameter that determines the number of filters in each convolutional layer. Our choices for filters in the consecutive layers were 16, 32, and 64 respectively. Zero-padding determines whether to pad the edges of the input to constrain the dimensions of the output of each layer. We did not use any zero padding because our sampling rate was high enough that a few points near the edges can be discarded. It may be necessary to use zero-padding for deeper networks where more information could be lost. We used kernel sizes of 16, 8, and 8 for the convolutional layers and 4 for all the pooling layers. Stride, which measures the distance between

the receptive fields, was chosen to be 1 for all the convolution layers and 4 for all the pooling layers. Dilation determines the overall size of each receptive field (which could be larger than the kernel size by having gaps in between). Here, it is a measure of the temporal extend of the convolutions. We observed that using dilation of 4 in the final two convolution layers improved the accuracy of the predictions. Yet another hyperparameter is the initial learning rate, which was set to 0.001. The final layout of our classifier DNN is shown in Figure 5.

A loss function (also known as a cost function) is required to compute the error after each iteration by measuring how close the outputs of the neural network are with respect to the target values. Common loss functions used for regression include mean absolute error and mean squared error. We chose the mean squared error function for the predictor. For classification, we used the cross-entropy loss function, which is the de facto standard [48].

Training Strategy

Over 300 hours of training was performed during the course of three weeks with most of the time spent on hyperparameter optimization. Once we chose the best performing DNNs, we trained it for a further 10 hours. Most of the intensive training was done on NVIDIA Tesla K40c, GTX 1080, and P100 GPUs. We relied on the new neural network functionality in version 11 of the Wolfram Language (Mathematica) [104]. This is based on the open-source MXNet [105] framework, which is optimized to take advantage of the CUDA deep learning library (cuDNN [41]) for GPU acceleration. A snapshot of the training process is shown in Figure 6. Moreover, the Wolfram Language was used for all the data processing and visualizations.

The stochastic gradient technique was used for our initial tests but we switched to the ADAM [66] method for the majority of our final training rounds as it provided the best results with the fastest training rate. Although we experimented with various regularization techniques, including L2 regularization [106], we determined that it was easier and faster to simply enhance the size of our training dataset by adding more simulations. It is expected that when considering the full parameter-space, regularization will be useful to reduce the size of the training set.

During this process, we developed a new strategy to improve the performance and reduce training times of the DNNs. By starting off the training with inputs having high SNR and then slowly increasing the amplitude of noise in each subsequent training session, we observed that the accuracy of prediction and classification can be maximized. We repeated this procedure a few times with the same DNN but different initial conditions to obtain the best results. We expect this new technique will be very useful for training neural networks, in general, with noisy data.

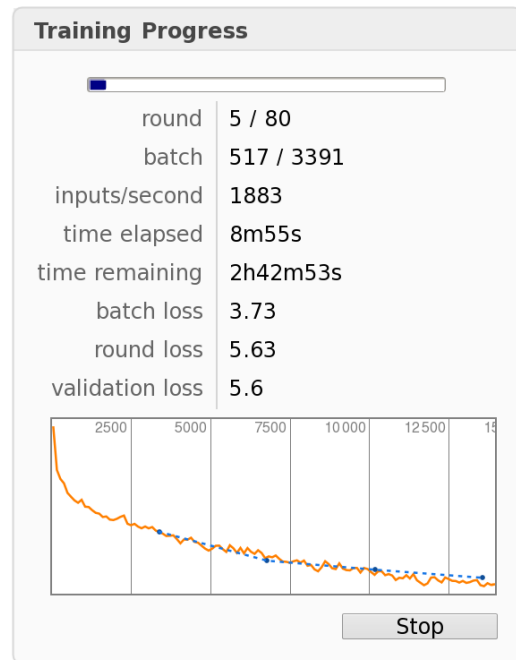


FIG. 6. This is a snapshot of one of our training sessions for parameter estimation, running on a Tesla K40c GPU, using version 11.0.1 of the Wolfram Language. The mean squared error on the training set is plotted in orange and the blue curve measures the error on the validation set.

V. RESULTS

We present the performance metrics of the classifier followed by the predictor in this section. The results we obtained with our DNN prototype are remarkable and provide a strong incentive to investigate more topologically complex DNNs for GW analysis which can be trained using real aLIGO strain data and templates of a broader range of GW signals including BNS inspirals, spin-aligned and spin-precessing BBH systems, eccentric compact binary coalescences, and supernovae.

Classifier Measurements

We created two versions of the classifier (A and B); version A has been tuned to minimize false detections while version B has been optimized to detect very weak signals at the cost of few false positives.

Classifier A achieves 100% accuracy for signals with $\text{SNR} \geq 0.6$ but only has an accuracy of 95% at $\text{SNR} = 0.5$. Confusion matrices of this classifier are shown in Figure 7. Classifier B was able to identify the presence of signals with an SNR as low as 0.5 with an accuracy of 99.8% or higher. The confusion matrix of this classifier at $\text{SNR} = 0.5$ is shown in the left panel of Figure 8. The accuracy of classifier B as a function of SNR is shown in the right panel of Figure 8 and its Accuracy Rejection Curve [107] (ARC) at a fixed SNR is shown in Figure 9.

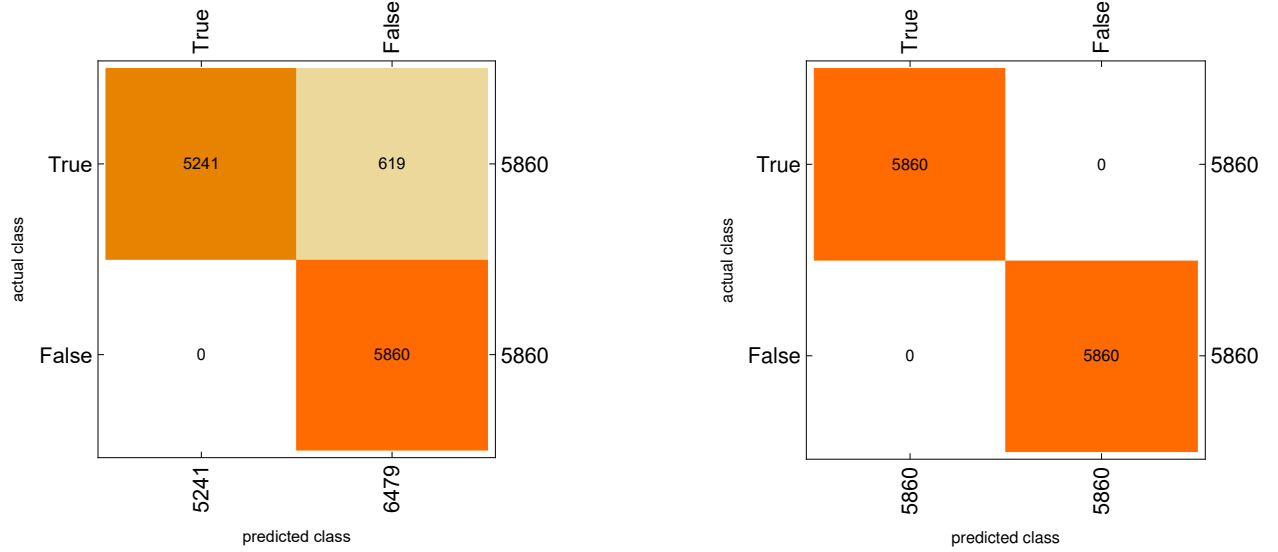


FIG. 7. Left panel: Confusion matrix of classifier A on a test set having $\text{SNR} = 0.5$ (the accuracy is 94.7%). Note that this classifier never obtains false positives. Right panel: Confusion matrix of classifier A on a test set having $\text{SNR} = 0.6$ (the accuracy is 100% for all signals with $\text{SNR} \geq 0.6$.)

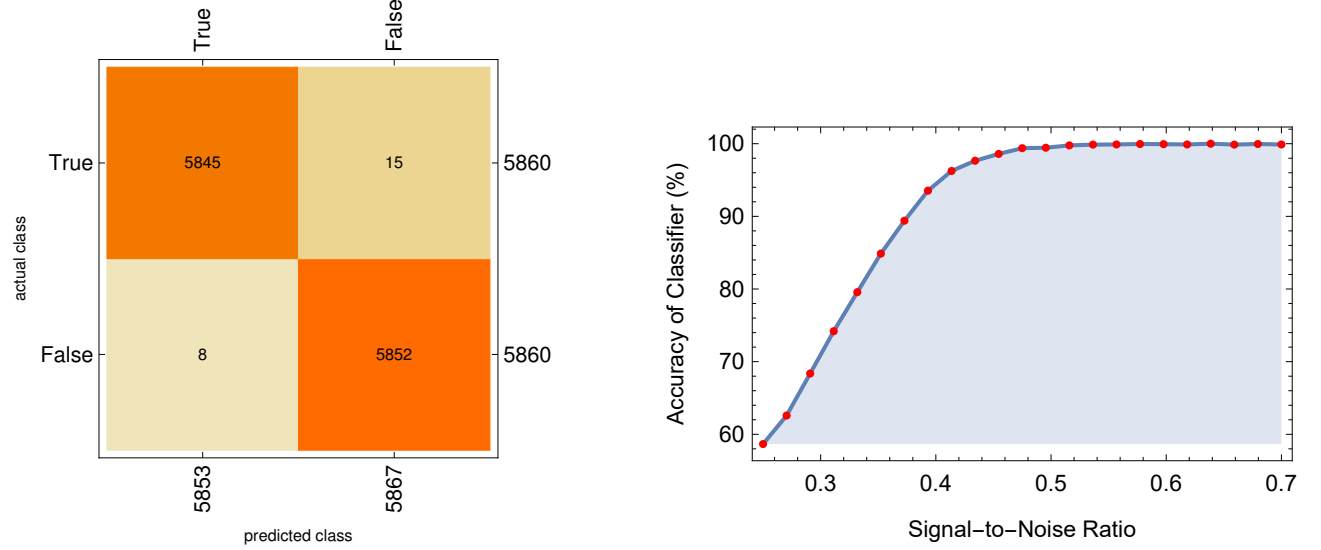


FIG. 8. Left panel: Confusion matrix of classifier B on a test set having $\text{SNR} = 0.5$ (the accuracy is 99.8%). Right panel: Accuracy of classifier B as a function of SNR. Unlike classifier A, the accuracy of classifier B at $\text{SNR} \geq 0.7$ saturates at approximately 99.89%, due to the constant rate of false detections.

These results demonstrate that the classifiers can be tuned to optimize for any aspect of detection, including false positive or false negative rates, and their rejection thresholds can be controlled. Furthermore, the output of the classifiers can be cross-validated by other signal processing methods to enhance the confidence levels.

Even though glitches and other non-Gaussian noise transients were not used during the training process, we observed that the classifier was able to correctly identify most of them as sources of noise rather than GW signals. By including templates of all types of known signals and

examples of transient detector noise in the training set, the classifier will be able to target a wider class of GW sources, while ignoring glitches automatically.

The classifier was also tested with a few NR simulations of eccentric BBH mergers, that we recently completed using the Einstein Toolkit [108] on Blue Waters, as well as several spin-aligned NR waveforms [109]. A sample of one of our eccentric waveforms is shown in Figure 10. The waveforms were scaled to have total mass between $120M_{\odot}$ to $150M_{\odot}$ to ensure that they were at least one second long. All these signals were detected with the

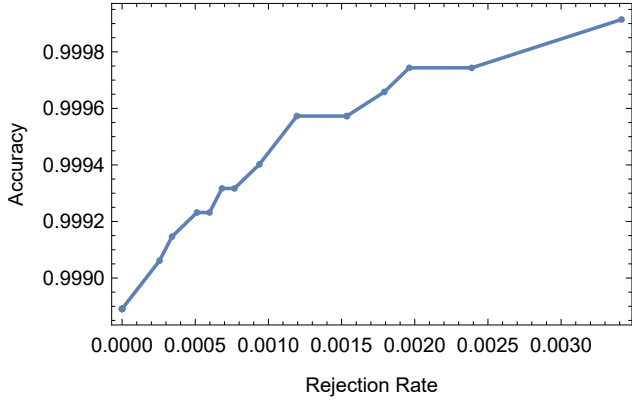


FIG. 9. This Accuracy Rejection Plot shows the trade-off between the accuracy and rejection rate of classifier B at SNR = 0.5. Therefore, the classifier can be tuned for higher accuracy at the cost of more false detections.

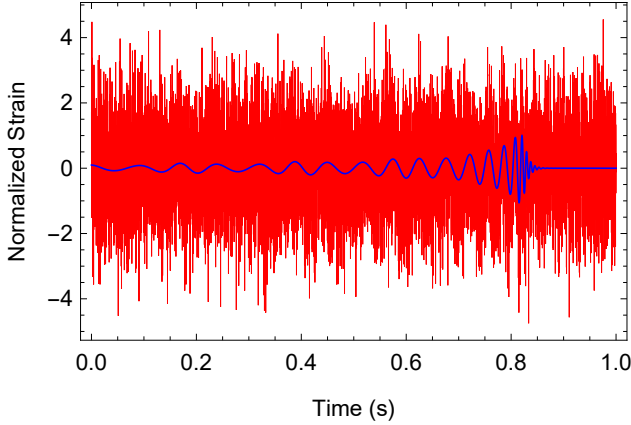


FIG. 10. This waveform was obtained from one of our eccentric NR simulations (B001) representing a BBH merger with equal component masses of $60M_{\odot}$, that has an initial eccentricity $e_0 = 0.076$ when it enters the aLIGO band, embedded in noisy data with SNR = 0.75. Our DNN pipeline successfully detected these signals with 100% accuracy and predicted the component masses with a mean relative error $\leq 11.3\%$.

same accuracy as the original test set. This is a very encouraging result, since recent analyses have made evident that existing LIGO algorithms used for source detection are not capable of accurately identifying and reconstructing the parameters of moderately eccentric signals [110–112]. This implies that DNN-based GW pipelines offer a natural framework to target a wide class of GW sources that are currently undetectable with existing detection algorithms.

For comparison, we trained all commonly used machine learning classifiers[104] — Random Forest, Support Vector Machine, k-Nearest Neighbors, Hidden Markov Model, Shallow ANNs, Naive Bayes, and Logistic Regression — with a smaller training set of 8000 elements. Unlike DNNs, none of these algorithms were able to directly handle raw noisy data as observed in Figure 11.

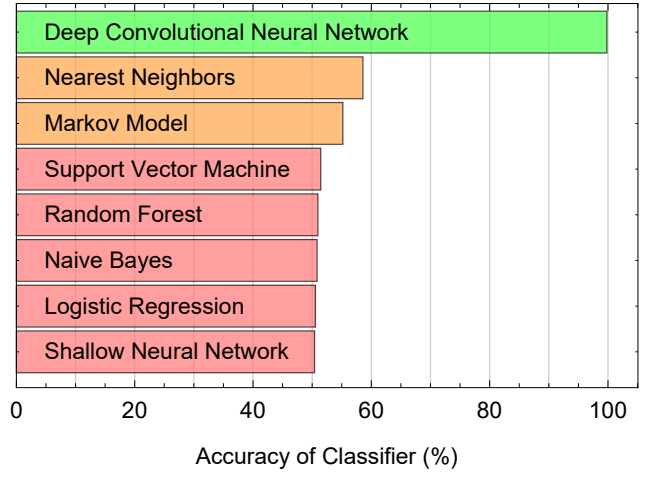


FIG. 11. The accuracy of different machine learning methods for the classification task is shown here. We used the same training and testing sets, half of which contained signals with fixed SNR = 0.6 and the other half being pure noise. The parameters for each model were chosen heuristically by the `Classify` function in the Wolfram Language [104]. Note that while the DNN reached an accuracy of 99.8%, the traditional machine learning algorithms could only obtain accuracies close to 50%, which means that they were randomly guessing.

Predictor Measurements

This is the very first demonstration that DNNs are capable of estimating multiple continuous parameters directly from noisy time-series inputs when trained with a template bank of expected signals.

Our predictor was able to successfully measure the masses of waveforms, that were not part of the training, with an error of the same order as the spacing between templates, for SNR ≥ 1.0 . This means that our algorithm was able to learn to effectively interpolate between the parameters that were shown to it in order to predict intermediate values.

The distribution of errors at a fixed SNR = 1 on our testing templates is shown in Figure 12 (right), where the mean relative error is 11.9%. For SNR between 0.5 and 1.0, the predictor provided estimates of the masses within 15% error. For extremely low SNR = 0.25, a mean error of approximately 47% was obtained. The variation in relative error against SNR is shown in Figure 12 (left).

Surprisingly, the predictor was also able to estimate the masses of our eccentric simulations with a mean relative error less than 14.5% for all SNR greater than 0.5. On testing with a few spin-aligned NR waveforms [109], we obtained a mean error $\leq 20\%$ for SNR ≥ 1 . The estimated values for the component masses of systems with non-zero spins were slightly lower than their true values. This is expected since spin-aligned waveforms have longer timespans than their non-spinning counterparts and they were not exposed to the DNN during the training stage.

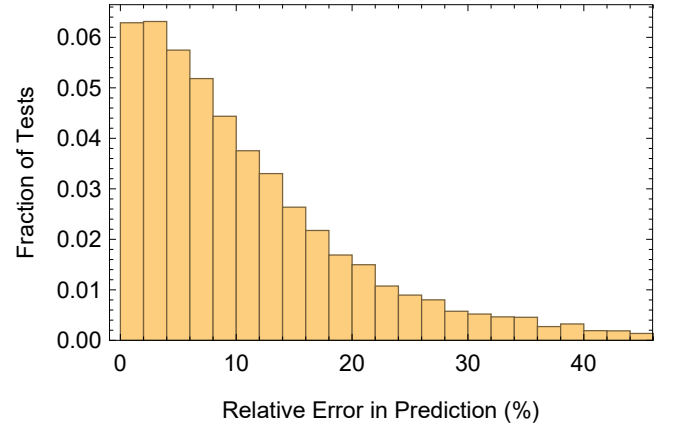
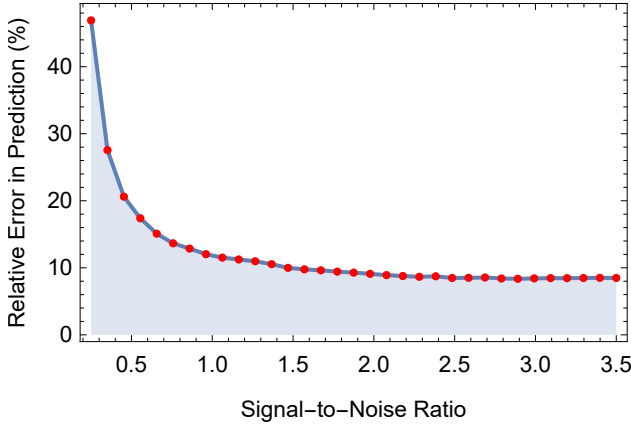


FIG. 12. Left panel: The mean percentage error of estimated masses with inputs at different SNR. Note that a relative error less than 47% was obtained for SNR = 0.25. The error at high SNR (about 10%) is proportional to the spacing between masses used in the training set. Right panel: The distribution of percentage error in the masses predicted on the test set at a fixed SNR = 1.0. The mean error is 11.9%.

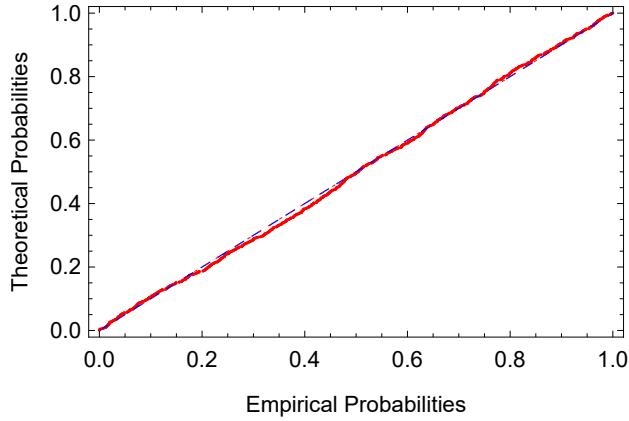


FIG. 13. This is a P-P (probability) plot of the distribution of errors in predicting m_1 in the region of parameter-space around $m_1 = 42M_\odot$ and $m_2 = 33M_\odot$. The estimated distribution is a Gaussian distribution with mean $= -0.56M_\odot$ and standard deviation $= 1.72M_\odot$. The errors have similar Gaussian distributions in other regions of the parameter-space.

While it is not possible to estimate theoretical uncertainties for the predictions made by a trained DNN, one can estimate the distribution of errors *empirically* at each region of the parameter-space. We observed that the errors always follow Gaussian normal distributions for the parameters that we are considering. A standard Probability-Probability (P-P) plot at a randomly chosen section of the parameter-space is shown in Figure 13.

After experimenting with common machine learning techniques [104] including Linear Regression, k-Nearest Neighbors, Shallow ANNs, Gaussian Process Regression, and Random Forest, we observed that, unlike DNNs, they could not predict even a single parameter (mass-ratio at fixed total mass) accurately as evident from Figure 14. This is expected since prediction is a harder problem than classification and traditional methods are

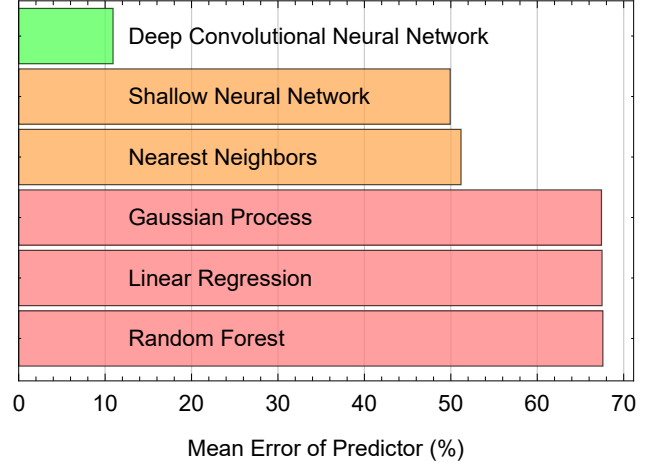


FIG. 14. This is the relative error obtained by different machine learning algorithms for predicting a single parameter, mass-ratio, using a training set containing 8000 elements with SNR = 0.6. The optimal parameters of each model was automatically determined by the `Predict` function in the Wolfram Language [104]. Note that a mean relative error of 52% can be achieved by simply guessing the same value of 3.2 for every input. Furthermore, extending these methods to multiple parameters is often not straightforward, whereas it simply involves adding an extra neuron to the DNN.

unable to deal directly with noisy time-series data.

Summary

The DNNs that we designed were able to accurately recognize and estimate parameters of gravitational waveforms, with extremely low SNR ≥ 0.25 , that were not included in the training set. Moreover, the classifier was able to identify a variety of simulated glitches as noise.

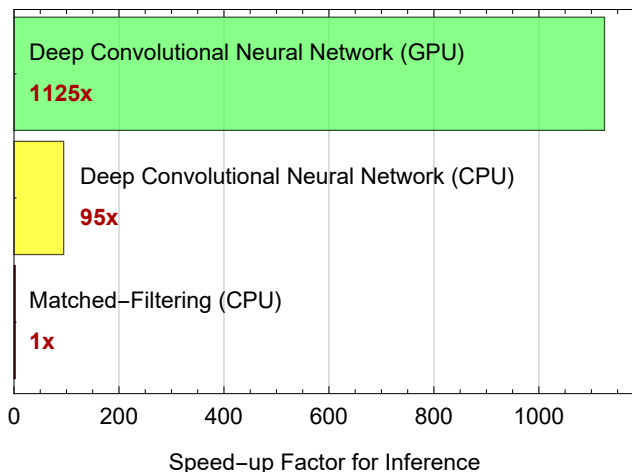


FIG. 15. The DNN-based pipeline is at least 1125 times faster on a GPU and 95 times faster on a CPU compared to matched-filtering when using the same template bank of 4000 waveforms (tested with an Intel Core i7-6500U CPU and an NVIDIA GeForce GTX 1080 GPU). More importantly, the evaluation time of a DNN is constant regardless of the size of training data, whereas the time taken for matched-filtering is proportional to the number of templates being considered. Therefore, when scaling to a template bank of 8000 waveforms we expect the speed-up factors to double.

The DNNs outperformed all traditional machine learning methods by a large margin when dealing directly with noisy time-series inputs for both classification and parameter estimation.

Although our training set did not contain waveforms from eccentric and spinning binaries, the performance of the predictor and classifier were excellent on these signals. This suggests that DNN-based pipelines are ideal tools to detect GW signals that would otherwise go unnoticed with current GW detection algorithms [110–112].

The average time taken for evaluating the classifier and predictor on inputs of 1 second duration is about 6.7 milliseconds and 0.6 milliseconds using a single CPU and GPU respectively. For comparison, we estimated the evaluation time for a standard implementation of the time-domain matched-filtering algorithm based on cross-correlations with the same template bank; the results are shown in Figure 15. Employing a GPU inference engine [113] (e.g. NVIDIA TensorRT) can further speed up the evaluation of data in bulk by several orders of magnitude. Thus, the compact size of the trained DNNs (4MB) combined with fast inference rates makes real-time analysis possible with minimal resources compared to existing methods.

Considering that all the neural networks we tested are

tiny by modern standards⁴, we believe that much higher accuracies over a wider range of parameters and types of signals can be obtained by exploring more complex configurations of neural networks, choosing more optimal hyperparameters, and using a larger set of carefully-chosen training templates.

VI. DISCUSSION

We have demonstrated that DNNs are powerful, yet computationally efficient, tools for time-domain signal processing and that they are particularly suited for both GW detection and characterization. We exposed the neural network to examples of modeled waveforms and allowed it to develop its own strategies to extract a variety of GW signals from noisy data. Our initial DNN was trained to predict only the mass-ratio at a fixed total mass and extending it to predict component masses was as simple as adding an extra neuron to the output layer. This strongly suggests that training the DNN to predict any number of parameters would be similarly straightforward.

Furthermore, our algorithm requires minimal pre-processing. In principle, aLIGO’s observed colored noise can be superimposed into our training set of GW templates, although we have used simulated white noise as a starting point here. It has been previously found that deep CNNs are capable of automatically learning to perform band-pass filtering on raw time-series inputs [114] and that they are excellent at suppressing highly non-stationary colored noise [115] especially when incorporating real-time noise characteristics [116]. This suggests that manually devised pre-processing and whitening steps may be completely eliminated in the future.

Big Data and Innovative Hardware

Deep learning is known for its scalability, i.e., the ability to take advantage of petascale or larger datasets. Training with billions of templates is already feasible with current technology. This data-driven approach allows us to easily scale to any number of parameters or to new types of signals or noises. As our theoretical understanding of scientific phenomena improves, one can add more categories of simulated waveforms with higher accuracy to our training set without modifying the underlying architecture of the DNNs.

Our fully trained DNN is only 4MB in size and encodes all the necessary information from the 4000 GW templates (350 megabytes) used for training. This shows

⁴ State-of-the-art DNNs used for commercial applications today contain hundreds of layers and are typically trained over periods ranging from a few weeks to months with petabytes of data on multiple high-performance GPUs optimized for deep learning.

that the DNNs have extracted and compressed the relevant information by learning patterns within the training data. Once trained, these DNNs can perform analysis within milliseconds. Evaluation can be considerably sped up by processing bulk data on GPUs.

Anticipating the growth of AI, many chip manufacturers are heavily investing in developing specialized hardware for training and evaluating DNNs. Unlike traditional CPUs, GPUs are far more power efficient and compact while being exponentially faster for deep learning. Several deep-learning-optimized GPUs are available including NVIDIA Titan X and P100. An ideal choice would be the new NVIDIA DGX-1 supercomputer dedicated for deep learning and AI accelerated analytics, one of which is located at the LIGO Hanford site. Having similar machines at LIGO Livingston and at Cascina for the advanced Virgo detector, will enable individually calibrated on-site GW searches. FPGAs are another promising development that can further accelerate DNNs [42].

DNNs can be utilized in any area of signal processing where large datasets are available. For example, this method can be applied to process raw image data from telescopes in real-time. Furthermore, the methodology that we have initiated in this article on combining numerical simulations with AI algorithms can allow us to efficiently exploit observational instruments (telescopes, astro-particle detectors, and other measurement devices), traditional supercomputers (and future exascale machines), as well as powerful emerging hardware architectures including GPUs, FPGAs, application-specific integrated circuit (ASICs), and brain-like chips (e.g. IBM's TrueNorth [117]), thus opening the door to a wealth of new science opportunities.

Coincident Detection of GWs and EM Counterparts

Our DNN pipeline can be extended to other types of signals by adding their templates to the training set. BNS inspirals and NSBH mergers are prime candidates for the central engines of short gamma ray bursts (sGRBs) [23–26]. The observational confirmation of this hypothesis [27–29] within the next few years will require coincident analyses of GWs, broadband EM follow-ups and astro-particle physics. Similarly, GWs emitted by rapidly rotating hypernovae — possible progenitors of long duration GRBs, collapsars, etc. — that are detected in coincidence with EM observations will provide unique insights into dynamics occurring in collapsing massive stars that would otherwise remain inaccessible [118, 119].

Having catalogs of state-of-the-art NR simulations of the expected gravitational waveforms of these events [93, 120–131] will be crucial to further train DNN-based pipelines to extend the depth of GW searches. In addition, DNNs are particularly suited for image and video processing, therefore, they can be trained to simultaneously search for the EM counterparts of these GW sources from raw image data obtained with telescopes.

New Science with Real-time Data Processing

The estimates of the astrophysical properties of GW candidates returned by our DNN predictor can be used as a starting point to speed up more informative Bayesian parameter estimation methods, while providing rough estimates to start instant EM follow-up campaigns.

Furthermore, DNNs can be applied to rapidly identify EM transients in raw image data from state-of-the-art and next-generation astronomical facilities. We are experimenting with DES data to train DNNs for this type of analysis. On-site GPU/FPGA based systems can be placed at telescope locations to accelerate EM follow-ups of GW candidates. If the identification of an EM transient can be carried out quickly, we can interface this information with a DNN-based GW detection pipeline and vice-versa. Joint analyses of this nature will enable real-time multimessenger astrophysics searches.

Deep learning methods can be immediately applied through distributed computing. Therefore, they would aid large-scale projects such as Einstein@Home [132] and SETI@Home [133]. Several open-source deep learning libraries, including MXNet, allow scalable distributed training and evaluation of neural networks simultaneously on heterogeneous portable devices, including smartphones and tablets, thus facilitating citizen science campaigns.

DNNs for Future Gravitational Wave Missions

Recent work suggests that space-based GW detectors such as the evolved Laser Interferometer Space Antenna (eLISA) [134, 135] will be able to detect stellar mass BBH systems weeks before they merge in the frequency band of ground-based GW detectors [136]. DNNs can be used to detect these sources in the eLISA and aLIGO frequency bands using a unified pipeline (on-board analysis may be possible in space with extremely power-efficient chips dedicated for deep learning inference). Furthermore, by training similar DNN-based pipelines with DES, LSST, WFIRST, and other open data, one may develop robust, low-latency classification algorithms to search for EM transients in the anticipated sky region where these events are expected to occur.

Another exciting prospect for GW detection is related to sources emitting in the nanoHertz region of the GW spectrum [137, 138], e.g., supermassive black hole binaries (SMBHBs), cosmic strings, and a stochastic GW background generated by a cosmological population of SMBHBs [139–144]. The expected signature of these events is under investigation using analytical [145–147] and statistical frameworks [148, 149]. DNNs can expand the parameter-space used in these searches, and provide a unified framework to simultaneously target these sources. Furthermore, DNNs may enable deep systematic GW searches taking into account correlations among pulsars in a detector network.

In summary, the flexibility and computational efficiency of DNNs could promote them as standard tools for multimessenger astrophysics.

Scope for Improvements

One may construct a multi-dimensional template bank based on state-of-the-art semi-analytical models and all available NR-based waveforms. Thereafter, one can superimpose samples of real aLIGO noise (and non-Gaussian transients) on these templates and carry out an intensive training procedure. Once this process is finished, the DNN may be used for real-time classification and parameter estimation while being periodically re-trained with more gravitational waveforms and recent aLIGO noise.

Out-of-core training techniques may be needed if the training data is too large to fit in RAM. The size of the data can be reduced by figuring out more optimal placements of templates within the parameter-space. Nonetheless, the DNNs have been observed to perform monotonically better when trained with increasing amounts of data [48]. An alternative would be to incorporate noise addition into the initial layers of the DNN so that the training set is smaller, containing only the clean signal templates, and new noise will be superimposed automatically in each iteration.

Our DNNs contain only 15 hidden layers in total. Developing efficient architectures of DNNs, that can be trained quickly and evaluated faster, is an active area of research. For example, GoogLeNet [85] is a very deep neural network that is an order of magnitude smaller in size and evaluates inputs much faster than competitors while achieving the same or higher levels of accuracy. Residual nets or ResNets [87] are the current state-of-the-art models for image processing and have been shown to monotonically improve when adding more layers.

CNNs are limited by the fact that they can only use fixed length tensors as inputs and outputs. On the other hand, Recurrent Neural Networks (RNNs), the deepest of all neural networks, have cyclic internal structures and are well-suited for time-series analysis since they can make decisions based on a stream of inputs rather than a vector of fixed length [49]. A powerful type of RNN called LSTM for Long-Short-Term-Memory (discovered in 1997 by Hochreiter and Schmidhuber [150]) is capable of remembering long-term dependencies in the input sequence. This suggests that there is a large scope for improvement in developing efficient architectures of DNNs for scientific data analysis.

Since the architecture of the classifier and predictor is almost identical, it may be possible to fuse their initial layers to minimize computational costs. Furthermore, hierarchical multi-task learning is possible with DNNs [88, 89], thus allowing a single network to classify inputs into categories and sub-categories, while also performing parameter estimation for each type of signal.

Stacking chunks of time-series data to produce multi-dimensional tensors can facilitate processing massive quantities of data efficiently on modern hardware, for example, to find signals that are very long in duration like inspirals of neutron stars. Experimenting with different loss functions (e.g. mean squared *relative* error) may improve the accuracy in certain regions of the parameter-space. The accuracy of the DNNs can be further enhanced by training an ensemble of different models and taking an average of the results for each input [48].

Due to the fast computation speed, low power consumption, and portability, it will be far more efficient to perform real-time analysis by directly sending a continuous stream of data to dedicated GPU inference engines at the aLIGO site to maximize the detection rate and minimize lags due to data transfer. Having powerful deep-learning-optimized machines at each detector will make it possible to continuously re-train the DNNs with the latest aLIGO data thus recalibrating them in real-time based on the characteristics of the current noise.

aLIGO uses a variety of independent sensors to monitor the environment and assess data quality. There are independent algorithms to estimate periods which must be vetoed due to disturbances that lead to a loss in detector sensitivity. It may be possible to train DNNs to perform this process of vetoing by taking into account simultaneous inputs from all sensors to determine data quality.

Work in Progress

We are currently working on using long stretches of real aLIGO data to re-train our DNNs by adding injections of all available templates of gravitational waveforms from binary systems, including inspirals of BNSs and stellar mass BBHs. We will directly use noise with a realistic PSD, accounting for real-time detector sensitivity while training the classifier to identify new categories of glitches and other non-Gaussian sources of noise, that closely mimic signals, in conjunction with unsupervised learning techniques for anomaly detection.

We are also performing large-scale fully automated simulations on Blue Waters for BBH systems with significant orbital eccentricity. This will provide a catalog of waveforms that extend the parameter-space beyond what aLIGO pipelines can currently cover.

VII. CONCLUSION

We have presented a novel framework for signal processing that is tailored to enable real-time multimessenger astrophysics, which differs significantly from existing scientific data analysis techniques in terms of both performance and scalability. We have provided a proof of concept which strongly indicates that DNNs are ideal tools to broaden the scope of current GW searches with

aLIGO and future GW missions. In practice, we envision using millions, or even billions, of GW templates to train DNNs for searches that target a wide class of GW transients ranging from GW bursts, compact binary coalescences, supernovae, and probable progenitors of EM transients. Furthermore, deeper layers can be used to encode as much astrophysical information as needed and multi-task learning can unify detection and classification of sources and glitches as well as parameter estimation under a single DNN pipeline.

Training DNNs efficiently requires powerful modern hardware, such as GPUs or FPGAs, and may take several days. This can be immediately achieved using aLIGO’s computational facilities, especially the NVIDIA DGX-1 server, purpose-built for deep learning workloads, located at the Hanford LIGO Lab. However, once a DNN is trained at a given aLIGO PSD, it would take only a few minutes to re-train it with the most recent PSD during a detection campaign. Therefore, it would be feasible to continuously re-calibrate the DNNs in real-time with the latest noise characteristics of each detector. This feature is of great relevance for GW searches, since we will be able to incorporate low-latency data quality information in DNN-based pipelines.

The entire DNN has a compact size (few megabytes) and, once trained, evaluation takes only milliseconds with a single CPU and microseconds with a GPU. This means that with the methods presented in this article, real-time GW searches can be carried out using an average laptop computer, while big datasets can be processed rapidly in bulk with GPU hardware dedicated for inference.

Employing DNNs for multimessenger astrophysics offers unprecedented opportunities to harness hyper-scale AI computing with emerging hardware architectures and cutting-edge software developed by industries. Nonetheless, the use of future exascale supercomputing facilities

will be critical for performing improved NR simulations that faithfully encode the gravitational and EM signatures of GW sources, which will be used to *teach* these intelligent algorithms. We expect that this novel approach will percolate in the scientific community since it provides an ideal framework to enable real-time multimessenger astrophysics and allows GW enthusiasts to participate in citizen science campaigns using personal computers or smartphone devices, once aLIGO data becomes open. We also anticipate that our new methodology for processing signals in noisy data will prove to be invaluable in many other areas of engineering, science, and technology. Therefore, this work is laying the foundations to seamlessly integrate diverse domains of expertise to enable and accelerate scientific discovery.

ACKNOWLEDGMENTS

This research is part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications. The eccentric numerical relativity simulations used in this article were performed with the open source, community software the Einstein Toolkit.

We express our gratitude to Gabrielle Allen, Ed Seidel, Roland Haas, Miguel Holgado, Haris Markakis, Justin Schive, and Prannoy Mupparaju for insightful comments and interactions. We thank Vlad Kindratenko for granting us access to several NVIDIA Tesla, GeForce, and P100 GPUs in the Innovative Systems Lab at NCSA. We also acknowledge Wolfram Research for developing the software stack used to carry out this analysis and to draft this publication.

-
- [1] B. P. Abbott, R. Abbott, T. D. Abbott, M. R. Abernathy, F. Acernese, K. Ackley, C. Adams, T. Adams, P. Addesso, R. X. Adhikari, and et al., *Physical Review Letters* **116**, 061102 (2016), [arXiv:1602.03837 \[gr-qc\]](#).
 - [2] The LIGO Scientific Collaboration and the Virgo Collaboration, ArXiv e-prints (2016), [arXiv:1606.04855 \[gr-qc\]](#).
 - [3] The LIGO Scientific Collaboration, the Virgo Collaboration, B. P. Abbott, R. Abbott, T. D. Abbott, M. R. Abernathy, F. Acernese, K. Ackley, C. Adams, T. Adams, and et al., ArXiv e-prints (2016), [arXiv:1606.04856 \[gr-qc\]](#).
 - [4] The LIGO Scientific Collaboration and the Virgo Collaboration, ArXiv e-prints (2016), [arXiv:1602.03840 \[gr-qc\]](#).
 - [5] K. Belczynski, M. Dominik, T. Bulik, R. O’Shaughnessy, C. Fryer, and D. E. Holz, *Astrophys. J. Lett* **715**, L138 (2010), [arXiv:1004.0386 \[astro-ph.HE\]](#).
 - [6] F. Antonini, S. Chatterjee, C. L. Rodriguez, M. Morscher, B. Pattabiraman, V. Kalogera, and F. A. Rasio, *Astrophys. J.* **816**, 65 (2016), [arXiv:1509.05080](#).
 - [7] B. P. Abbott, R. Abbott, T. D. Abbott, M. R. Abernathy, F. Acernese, K. Ackley, C. Adams, T. Adams, P. Addesso, R. X. Adhikari, and et al., *Astrophys. J. Lett* **818**, L22 (2016), [arXiv:1602.03846 \[astro-ph.HE\]](#).
 - [8] B. P. Abbott, R. Abbott, T. D. Abbott, M. R. Abernathy, F. Acernese, K. Ackley, C. Adams, T. Adams, P. Addesso, R. X. Adhikari, and et al., *Living Reviews in Relativity* **19** (2016), 10.1007/lrr-2016-1, [arXiv:1304.0670 \[gr-qc\]](#).
 - [9] C. L. Rodriguez, S. Chatterjee, and F. A. Rasio, *Phys. Rev. D* **93**, 084029 (2016), [arXiv:1602.02444 \[astro-ph.HE\]](#).
 - [10] K. Belczynski, D. E. Holz, T. Bulik, and R. O’Shaughnessy, *Nature (London)* **534**, 512 (2016), [arXiv:1602.04531 \[astro-ph.HE\]](#).
 - [11] P. Marchant, N. Langer, P. Podsiadlowski, T. M.

- Tauris, and T. J. Moriya, *A&A* **588**, A50 (2016), [arXiv:1601.03718 \[astro-ph.SR\]](#).
- [12] S. E. de Mink and I. Mandel, *MNRAS* **460**, 3545 (2016), [arXiv:1603.02291 \[astro-ph.HE\]](#).
- [13] The LIGO Scientific Collaboration, the Virgo Collaboration, B. P. Abbott, R. Abbott, T. D. Abbott, M. R. Abernathy, F. Acernese, K. Ackley, C. Adams, T. Adams, and et al., ArXiv e-prints (2016), [arXiv:1606.01262 \[gr-qc\]](#).
- [14] The LIGO Scientific Collaboration, the Virgo Collaboration, B. P. Abbott, R. Abbott, T. D. Abbott, M. R. Abernathy, F. Acernese, K. Ackley, C. Adams, T. Adams, and et al., ArXiv e-prints (2016), [arXiv:1611.07531 \[gr-qc\]](#).
- [15] The LIGO Scientific Collaboration, J. Aasi, et al., *Classical and Quantum Gravity* **32**, 074001 (2015), [arXiv:1411.4547 \[gr-qc\]](#).
- [16] F. Acernese et al., *Classical and Quantum Gravity* **32**, 024001 (2015), [arXiv:1408.3978 \[gr-qc\]](#).
- [17] E. Hirose, T. Sekiguchi, R. Kumar, R. Takahashi, and the KAGRA Collaboration, *Classical and Quantum Gravity* **31**, 224004 (2014).
- [18] C. S. Unnikrishnan, *International Journal of Modern Physics D* **22**, 1341010 (2013).
- [19] The Dark Energy Survey Collaboration, ArXiv Astrophysics e-prints (2005), [astro-ph/0510346](#).
- [20] LSST Science Collaboration, P. A. Abell, J. Allison, S. F. Anderson, J. R. Andrew, J. R. P. Angel, L. Armus, D. Arnett, S. J. Asztalos, T. S. Axelrod, and et al., ArXiv e-prints (2009), [arXiv:0912.0201 \[astro-ph.IM\]](#).
- [21] L. Amendola, S. Appleby, D. Bacon, T. Baker, M. Baldi, N. Bartolo, A. Blanchard, C. Bonvin, S. Borgani, E. Branchini, C. Burrage, S. Camera, C. Carbone, L. Casarini, M. Cropper, C. de Rham, C. Di Porto, A. Ealet, P. G. Ferreira, F. Finelli, J. García-Bellido, T. Giannantonio, L. Guzzo, A. Heavens, L. Heisenberg, C. Heymans, H. Hoekstra, L. Hollenstein, R. Holmes, O. Horst, K. Jahnke, T. D. Kitching, T. Koivisto, M. Kunz, G. La Vacca, M. March, E. Majerotto, K. Markovic, D. Marsh, F. Marulli, R. Massey, Y. Mellier, D. F. Mota, N. Nunes, W. Percival, V. Pettorino, C. Porciani, C. Quercellini, J. Read, M. Rinaldi, D. Sapone, R. Scaramella, C. Skordis, F. Simpson, A. Taylor, S. Thomas, R. Trotta, L. Verde, F. Vernizzi, A. Vollmer, Y. Wang, J. Weller, and T. Zlosnik, *Living Reviews in Relativity* **16** (2013), 10.12942/lrr-2013-6, [arXiv:1206.1225](#).
- [22] D. Spergel, N. Gehrels, J. Breckinridge, M. Donahue, A. Dressler, B. S. Gaudi, T. Greene, O. Guyon, C. Hirata, J. Kalirai, N. J. Kasdin, W. Moos, S. Perlmutter, M. Postman, B. Rauscher, J. Rhodes, Y. Wang, D. Weinberg, J. Centrella, W. Traub, C. Baltay, J. Colbert, D. Bennett, A. Kiessling, B. Macintosh, J. Merten, M. Mortonson, M. Penny, E. Rozo, D. Savransky, K. Stapelfeldt, Y. Zu, C. Baker, E. Cheng, D. Content, J. Dooley, M. Foote, R. Goullioud, K. Grady, C. Jackson, J. Kruk, M. Levine, M. Melton, C. Peddie, J. Ruffa, and S. Shaklan, ArXiv e-prints (2013), [arXiv:1305.5422 \[astro-ph.IM\]](#).
- [23] D. Eichler, M. Livio, T. Piran, and D. N. Schramm, *Nature (London)* **340**, 126 (1989).
- [24] B. Paczynski, *Astrophys. J. Lett* **308**, L43 (1986).
- [25] R. Narayan, B. Paczynski, and T. Piran, *Astrophys. J. Lett* **395**, L83 (1992), [astro-ph/9204001](#).
- [26] C. S. Kochanek and T. Piran, *Astrophys. J.* **417**, L17 (1993), [arXiv:astro-ph/9305015 \[astro-ph\]](#).
- [27] T. Piran, E. Nakar, and S. Rosswog, *MNRAS* **430**, 2121 (2013), [arXiv:1204.6242 \[astro-ph.HE\]](#).
- [28] W. H. Lee, E. Ramirez-Ruiz, and G. van de Ven, *Astrophys. J.* **720**, 953 (2010).
- [29] W. H. Lee and E. Ramirez-Ruiz, *New Journal of Physics* **9**, 17 (2007), [astro-ph/0701874](#).
- [30] C. Röver, M.-A. Bizouard, N. Christensen, H. Dimmelmeier, I. S. Heng, and R. Meyer, *Phys. Rev. D* **80**, 102004 (2009), [arXiv:0909.1093 \[gr-qc\]](#).
- [31] T. B. Littenberg, B. Farr, S. Coughlin, and V. Kalogera, *Astrophys. J.* **820**, 7 (2016), [arXiv:1601.02661 \[astro-ph.HE\]](#).
- [32] S. A. Usman, A. H. Nitz, I. W. Harry, C. M. Biwer, D. A. Brown, M. Cabero, C. D. Capano, T. Dal Canton, T. Dent, S. Fairhurst, M. S. Kehl, D. Keppel, B. Krishnan, A. Lenon, A. Lundgren, A. B. Nielsen, L. P. Pekowsky, H. P. Pfeiffer, P. R. Saulson, M. West, and J. L. Willis, *Classical and Quantum Gravity* **33**, 215004 (2016), [arXiv:1508.02357 \[gr-qc\]](#).
- [33] S. Klimenko, I. Yakushin, M. Rakhmanov, and G. Mitselmakher, *Classical and Quantum Gravity* **21**, S1685 (2004), [gr-qc/0407025](#).
- [34] S. Klimenko and G. Mitselmakher, *Classical and Quantum Gravity* **21**, S1819 (2004).
- [35] S. Klimenko, S. Mohanty, M. Rakhmanov, and G. Mitselmakher, *Phys. Rev. D* **72**, 122002 (2005), [gr-qc/0508068](#).
- [36] S. Klimenko, I. Yakushin, A. Mercer, and G. Mitselmakher, *Classical and Quantum Gravity* **25**, 114029 (2008), [arXiv:0802.3232 \[gr-qc\]](#).
- [37] S. Klimenko, G. Vedovato, M. Drago, G. Mazzolo, G. Mitselmakher, C. Pankow, G. Prodi, V. Re, F. Salemi, and I. Yakushin, *Phys. Rev. D* **83**, 102001 (2011), [arXiv:1101.5408 \[astro-ph.IM\]](#).
- [38] N. J. Cornish and T. B. Littenberg, *Classical and Quantum Gravity* **32**, 135012 (2015), [arXiv:1410.3835 \[gr-qc\]](#).
- [39] S. Klimenko, G. Vedovato, M. Drago, F. Salemi, V. Tiwari, G. A. Prodi, C. Lazzaro, K. Ackley, S. Tiwari, C. F. Da Silva, and G. Mitselmakher, *Phys. Rev. D* **93**, 042004 (2016), [arXiv:1511.05999 \[gr-qc\]](#).
- [40] V. Tiwari, S. Klimenko, N. Christensen, E. A. Huerta, S. R. P. Mohapatra, A. Gopakumar, M. Haney, P. Ajith, S. T. McWilliams, G. Vedovato, M. Drago, F. Salemi, G. A. Prodi, C. Lazzaro, S. Tiwari, G. Mitselmakher, and F. Da Silva, *Phys. Rev. D* **93**, 043007 (2016), [arXiv:1511.09240 \[gr-qc\]](#).
- [41] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer, ArXiv e-prints (2014), [arXiv:1410.0759](#).
- [42] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, in *Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, FPGA '15 (ACM, New York, NY, USA, 2015) pp. 161–170.
- [43] I. Harry, S. Privitera, A. Bohé, and A. Buonanno, *Phys. Rev. D* **94**, 024012 (2016), [arXiv:1603.02444 \[gr-qc\]](#).
- [44] C. Capano, Y. Pan, and A. Buonanno, ArXiv e-prints (2013), [arXiv:1311.1286 \[gr-qc\]](#).
- [45] D. J. C. Mackay, *Information Theory, Inference and Learning Algorithms*, by David J. C. MacKay, pp. 640. ISBN 0521642981. Cambridge, UK: Cambridge University Press, October 2003. (2003) p. 640.

- [46] C. J. Moore, C. P. L. Berry, A. J. K. Chua, and J. R. Gair, *Phys. Rev. D* **93**, 064001 (2016), [arXiv:1509.04066 \[gr-qc\]](#).
- [47] C. J. Moore and J. R. Gair, *Physical Review Letters* **113**, 251101 (2014), [arXiv:1412.3657 \[gr-qc\]](#).
- [48] I. Goodfellow, Y. Bengio, and A. Courville, “Deep learning,” (2016), book in preparation for MIT Press.
- [49] J. Schmidhuber, *Neural Networks* **61**, 85 (2015), published online 2014; based on TR [arXiv:1404.7828 \[cs.NE\]](#).
- [50] Y. Bengio, A. Courville, and P. Vincent, *IEEE Trans. Pattern Anal. Mach. Intell.* **35**, 1798 (2013).
- [51] Y. Lecun, Y. Bengio, and G. Hinton, *Nature* **521**, 436 (2015).
- [52] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, *ArXiv e-prints* (2016), [arXiv:1609.03499 \[cs.SD\]](#).
- [53] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, *Journal of Big Data* **2**, 1 (2015).
- [54] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, 1st ed. (MIT Press, Cambridge, MA, USA, 1998).
- [55] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, *Nature* **529**, 484 (2016), article.
- [56] J. Son Chung, A. Senior, O. Vinyals, and A. Zisserman, *ArXiv e-prints* (2016), [arXiv:1611.05358 \[cs.CV\]](#).
- [57] M. Nielsen, *Neural Networks and Deep Learning* (2016) e-book.
- [58] D. Graupe, *Principles of Artificial Neural Networks, 3rd edition*, pp. 500, ISBN 9789814522755. University of Illinois, Chicago, USA. World Scientific (2013).
- [59] F. Rosenblatt, *Psychological Review* **65**, 386 (1958).
- [60] M. Minsky and S. Papert, “Perceptrons : an introduction to computational geometry,” (1969).
- [61] K. Hornik, M. Stinchcombe, and H. White, *Neural Netw.* **2**, 359 (1989).
- [62] “Wikimedia Commons: Artificial Neural Network,” https://upload.wikimedia.org/wikipedia/commons/thumb/e/e4/Artificial_neural_network.svg/2000px-Artificial_neural_network.svg.png, accessed: 12-30-2016.
- [63] K. Jarrett, K. Kavukcuoglu, and Y. Lecun, “What is the best multi-stage architecture for object recognition?”.
- [64] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, in *Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop* (Springer-Verlag, London, UK, UK, 1998) pp. 9–50.
- [65] S. Ruder, *CoRR abs/1609.04747* (2016).
- [66] D. P. Kingma and J. Ba, *CoRR abs/1412.6980* (2014).
- [67] K. Fukushima, *Biological Cybernetics* **36**, 193 (1980).
- [68] Y. LeCun and Y. Bengio (MIT Press, Cambridge, MA, USA, 1998) Chap. Convolutional Networks for Images, Speech, and Time Series, pp. 255–258.
- [69] A. Krizhevsky, I. Sutskever, and G. E. Hinton, in *Advances in Neural Information Processing Systems 25*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Inc., 2012) pp. 1097–1105.
- [70] B. J. Owen and B. S. Sathyaprakash, *Phys. Rev. D* **60**, 022002 (1999), [gr-qc/9808076](#).
- [71] J. Veitch, V. Raymond, B. Farr, W. Farr, P. Graff, S. Vitale, B. Aylott, K. Blackburn, N. Christensen, M. Coughlin, W. Del Pozzo, F. Feroz, J. Gair, C.-J. Haster, V. Kalogera, T. Littenberg, I. Mandel, R. O’Shaughnessy, M. Pitkin, C. Rodriguez, C. Röver, T. Sidery, R. Smith, M. Van Der Sluys, A. Vecchio, W. Vousden, and L. Wade, *Phys. Rev. D* **91**, 042003 (2015), [arXiv:1409.7215 \[gr-qc\]](#).
- [72] P. Graff, F. Feroz, M. P. Hobson, and A. Lasenby, *MNRAS* **421**, 169 (2012), [arXiv:1110.2997 \[astro-ph.IM\]](#).
- [73] J. Powell, A. Torres-Forné, R. Lynch, D. Trifirò, E. Cuoco, M. Cavaglià, I. S. Heng, and J. A. Font, *ArXiv e-prints* (2016), [arXiv:1609.06262 \[astro-ph.IM\]](#).
- [74] J. Powell, D. Trifirò, E. Cuoco, I. S. Heng, and M. Cavaglià, *Classical and Quantum Gravity* **32**, 215012 (2015), [arXiv:1505.01299 \[astro-ph.IM\]](#).
- [75] M. Zevin, S. Coughlin, S. Bahaadini, E. Besler, N. Rohani, S. Allen, M. Cabero, K. Crowston, A. Katsagelos, S. Larson, T. K. Lee, C. Lintott, T. Littenberg, A. Lundgren, C. Oesterlund, J. Smith, L. Trouille, and V. Kalogera, *ArXiv e-prints* (2016), [arXiv:1611.04596 \[gr-qc\]](#).
- [76] A. Graves, A. Mohamed, and G. E. Hinton, *CoRR abs/1303.5778* (2013).
- [77] R. Pascanu, T. Mikolov, and Y. Bengio, in *ICML (3)*.
- [78] L. de Oliveira, M. Kagan, L. Mackey, B. Nachman, and A. Schwartzman, *Journal of High Energy Physics* **7**, 69 (2016), [arXiv:1511.05190 \[hep-ph\]](#).
- [79] A. Aurisano, A. Radovic, D. Rocco, A. Himmel, M. D. Messier, E. Niner, G. Pawloski, F. Psihas, A. Sousa, and P. Vahle, *Journal of Instrumentation* **11**, P09001 (2016), [arXiv:1604.01444 \[hep-ex\]](#).
- [80] P. Baldi, P. Sadowski, and D. Whiteson, *Nature Communications* **5**, 4308 EP (2014), article.
- [81] N. Mukund, S. Abraham, S. Kandhasamy, S. Mitra, and N. Sajeeth Philip, *ArXiv e-prints* (2016), [arXiv:1609.07259 \[astro-ph.IM\]](#).
- [82] T. J. O’Shea, J. Corgan, and T. C. Clancy, “Convolutional radio modulation recognition networks,” in *Engineering Applications of Neural Networks: 17th International Conference, EANN 2016, Aberdeen, UK, September 2-5, 2016, Proceedings*, edited by C. Jayne and L. Iliadis (Springer International Publishing, Cham, 2016) pp. 213–226.
- [83] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, “Time series classification using multi-channels deep convolutional neural networks,” in *Web-Age Information Management: 15th International Conference, WAIM 2014, Macau, China, June 16-18, 2014. Proceedings*, edited by F. Li, G. Li, S.-w. Hwang, B. Yao, and Z. Zhang (Springer International Publishing, Cham, 2014) pp. 298–310.
- [84] M. Zevin and Gravity Spy, in *American Astronomical Society Meeting Abstracts*, American Astronomical Society Meeting Abstracts, Vol. 228 (2016) p. 109.02.
- [85] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, *CoRR abs/1409.4842* (2014).
- [86] K. Simonyan and A. Zisserman, *CoRR abs/1409.1556* (2014).

- [87] K. He, X. Zhang, S. Ren, and J. Sun, **CoRR abs/1512.03385** (2015).
- [88] R. Caruana, in *Proceedings of the Tenth International Conference on Machine Learning* (Morgan Kaufmann, 1993) pp. 41–48.
- [89] T. Zeng and S. Ji, in *2015 IEEE International Conference on Data Mining* (2015) pp. 579–588.
- [90] D. Shoemaker, “Advanced LIGO anticipated sensitivity curves,” (2010), <https://dcc.ligo.org/cgi-bin/DocDB/ShowDocument?docid=2974>.
- [91] B. S. Sathyaprakash and B. F. Schutz, **Living Reviews in Relativity** **12**, 2 (2009), [arXiv:0903.0338 \[gr-qc\]](#).
- [92] A. H. Mroué, M. A. Scheel, B. Szilágyi, H. P. Pfeiffer, M. Boyle, D. A. Hemberger, L. E. Kidder, G. Lovelace, S. Ossokine, N. W. Taylor, A. Zenginoğlu, L. T. Buchman, T. Chu, E. Foley, M. Giesler, R. Owen, and S. A. Teukolsky, **Physical Review Letters** **111**, 241104 (2013), [arXiv:1304.6077 \[gr-qc\]](#).
- [93] F. Löffler, J. Faber, E. Bentivegna, T. Bode, P. Diener, R. Haas, I. Hinder, B. C. Mundim, C. D. Ott, E. Schnetter, G. Allen, M. Campanelli, and P. Laguna, **Classical and Quantum Gravity** **29**, 115001 (2012), [arXiv:1111.3344 \[gr-qc\]](#).
- [94] L. Blanchet, **Living Reviews in Relativity** **9**, 4 (2006).
- [95] S. A. Teukolsky, **Astrophys. J.** **185**, 635 (1973).
- [96] Y. Mino, M. Sasaki, M. Shibata, H. Tagoshi, and T. Tanaka, **Progress of Theoretical Physics Supplement** **128**, 1 (1997), [arXiv:gr-qc/9712057](#).
- [97] M. Sasaki and H. Tagoshi, **Living Reviews in Relativity** **6**, 6 (2003), [gr-qc/0306120](#).
- [98] A. Taracchini, A. Buonanno, Y. Pan, T. Hinderer, M. Boyle, D. A. Hemberger, L. E. Kidder, G. Lovelace, A. H. Mroué, H. P. Pfeiffer, M. A. Scheel, B. Szilágyi, N. W. Taylor, and A. Zenginoğlu, **Phys. Rev. D** **89**, 061502 (2014), [arXiv:1311.2544 \[gr-qc\]](#).
- [99] Y. LeCun, L. Bottou, G. B. Orr, and K. R. Müller, “Efficient backprop,” in *Neural Networks: Tricks of the Trade*, edited by G. B. Orr and K.-R. Müller (Springer Berlin Heidelberg, Berlin, Heidelberg, 1998) pp. 9–50.
- [100] A. Ng, “Machine learning yearning,” (2016), book in preparation.
- [101] S. Ioffe and C. Szegedy, **CoRR abs/1502.03167** (2015).
- [102] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, **Journal of Machine Learning Research** **15**, 1929 (2014).
- [103] J. Snoek, H. Larochelle, and R. P. Adams, in *Advances in Neural Information Processing Systems 25*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Inc., 2012) pp. 2951–2959.
- [104] “Wolfram Language System & Documentation Center,” <https://reference.wolfram.com/language/>, accessed: 2016-12-30.
- [105] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang, **CoRR abs/1512.01274** (2015).
- [106] A. Y. Ng, in *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML ’04 (ACM, New York, NY, USA, 2004) pp. 78–.
- [107] M. S. A. Nadeem, J.-D. Zucker, and B. Hanczar.
- [108] F. Löffler, J. Faber, E. Bentivegna, T. Bode, P. Diener, R. Haas, I. Hinder, B. C. Mundim, C. D. Ott, E. Schnetter, G. Allen, M. Campanelli, and P. Laguna, **Classical and Quantum Gravity** **29**, 115001 (2012), [arXiv:1111.3344 \[gr-qc\]](#).
- [109] T. Chu, H. Fong, P. Kumar, H. P. Pfeiffer, M. Boyle, D. A. Hemberger, L. E. Kidder, M. A. Scheel, and B. Szilágyi, ArXiv e-prints (2015), [arXiv:1512.06800 \[gr-qc\]](#).
- [110] E. A. Huerta, P. Kumar, B. Agarwal, D. George, H.-Y. Schive, H. P. Pfeiffer, R. Haas, W. Ren, T. Chu, M. Boyle, D. A. Hemberger, L. E. Kidder, M. A. Scheel, and B. Szilágyi, ArXiv e-prints (2016), [arXiv:1609.05933 \[gr-qc\]](#).
- [111] E. A. Huerta, P. Kumar, S. T. McWilliams, R. O’Shaughnessy, and N. Yunes, **Phys. Rev. D** **90**, 084016 (2014), [arXiv:1408.3406 \[gr-qc\]](#).
- [112] E. A. Huerta and D. A. Brown, **Phys. Rev. D** **87**, 127501 (2013), [arXiv:1301.1895 \[gr-qc\]](#).
- [113] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, **SIGARCH Comput. Archit. News** **44**, 243 (2016).
- [114] W. Dai, C. Dai, S. Qu, J. Li, and S. Das, **CoRR abs/1610.00087** (2016).
- [115] Y. Xu, J. Du, L. R. Dai, and C. H. Lee, **IEEE/ACM Transactions on Audio, Speech, and Language Processing** **23**, 7 (2015).
- [116] A. Kumar and D. Florêncio, **CoRR abs/1605.02427** (2016).
- [117] J. Sawada, F. Akopyan, A. S. Cassidy, B. Taba, M. V. Debole, P. Datta, R. Alvarez-Icaza, A. Amir, J. V. Arthur, A. Andreopoulos, R. Appuswamy, H. Baier, D. Barch, D. J. Berg, C. d. Nolfo, S. K. Esser, M. Flickner, T. A. Horvath, B. L. Jackson, J. Kunitz, S. Lekuch, M. Mastro, T. Melano, P. A. Merolla, S. E. Millman, T. K. Nayak, N. Pass, H. E. Penner, W. P. Risk, K. Schleupen, B. Shaw, H. Wu, B. Giera, A. T. Moody, N. Mundhenk, B. C. Van Essen, E. X. Wang, D. P. Widemann, Q. Wu, W. E. Murphy, J. K. Infantolino, J. A. Ross, D. R. Shires, M. M. Vindiola, R. Namburu, and D. S. Modha, in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC ’16 (IEEE Press, Piscataway, NJ, USA, 2016) pp. 12:1–12:12.
- [118] C. D. Ott, **Classical and Quantum Gravity** **26**, 063001 (2009), [arXiv:0809.0695](#).
- [119] E. S. Phinney, in *astro2010: The Astronomy and Astrophysics Decadal Survey*, Astronomy, Vol. 2010 (2009) [arXiv:0903.0098 \[astro-ph.CO\]](#).
- [120] F. Foucart, M. B. Deaton, M. D. Duez, L. E. Kidder, I. MacDonald, C. D. Ott, H. P. Pfeiffer, M. A. Scheel, B. Szilágyi, and S. A. Teukolsky, **Phys. Rev. D** **87**, 084006 (2013), [arXiv:1212.4810 \[gr-qc\]](#).
- [121] F. Foucart, D. Desai, W. Brege, M. D. Duez, D. Kasen, D. A. Hemberger, L. E. Kidder, H. P. Pfeiffer, and M. A. Scheel, ArXiv e-prints (2016), [arXiv:1611.01159 \[astro-ph.HE\]](#).
- [122] R. Haas, C. D. Ott, B. Szilágyi, J. D. Kaplan, J. Lipuner, M. A. Scheel, K. Barkett, C. D. Muhlberger, T. Dietrich, M. D. Duez, F. Foucart, H. P. Pfeiffer, L. E. Kidder, and S. A. Teukolsky, **Phys. Rev. D** **93**, 124062 (2016), [arXiv:1604.00782 \[gr-qc\]](#).
- [123] Z. B. Etienne, V. Paschalidis, R. Haas, P. Mösta, and S. L. Shapiro, **Classical and Quantum Gravity** **32**, 175009 (2015), [arXiv:1501.07276 \[astro-ph.HE\]](#).
- [124] P. Mösta, B. C. Mundim, J. A. Faber, R. Haas, S. C. Noble, T. Bode, F. Löffler, C. D. Ott, C. Reisswig, and E. Schnetter, **Classical and Quantum Gravity** **31**,

- 015005 (2014), [arXiv:1304.5544 \[gr-qc\]](#).
- [125] S. E. Gossan, P. Sutton, A. Stuver, M. Zanolin, K. Gill, and C. D. Ott, *Phys. Rev. D* **93**, 042002 (2016), [arXiv:1511.02836 \[astro-ph.HE\]](#).
 - [126] E. Schnetter, S. H. Hawley, and I. Hawke, *Classical and Quantum Gravity* **21**, 1465 (2004), [gr-qc/0310042](#).
 - [127] J. Thornburg, *Classical and Quantum Gravity* **21**, 743 (2004), [gr-qc/0306056](#).
 - [128] <http://einstein toolkit.org>.
 - [129] C. D. Ott, <http://sntheory.org>.
 - [130] T. Kuroda, K. Kotake, and T. Takiwaki, *Astrophys. J. Lett* **829**, L14 (2016), [arXiv:1605.09215 \[astro-ph.HE\]](#).
 - [131] H. Andresen, B. Mueller, E. Mueller, and H.-T. Janka, ArXiv e-prints (2016), [arXiv:1607.05199 \[astro-ph.HE\]](#).
 - [132] H. J. Pletsch and B. Allen, *Physical Review Letters* **103**, 181102 (2009), [arXiv:0906.0023 \[gr-qc\]](#).
 - [133] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer, *Commun. ACM* **45**, 56 (2002).
 - [134] P. Amaro-Seoane, S. Aoudia, S. Babak, P. Binétruy, E. Berti, A. Bohé, C. Caprini, M. Colpi, N. J. Cornish, K. Danzmann, J.-F. Dufaux, J. Gair, O. Jennrich, P. Jetzer, A. Klein, R. N. Lang, A. Lobo, T. Littenberg, S. T. McWilliams, G. Nelemans, A. Petiteau, E. K. Porter, B. F. Schutz, A. Sesana, R. Stebbins, T. Sumner, M. Vallisneri, S. Vitale, M. Volonteri, and H. Ward, *Classical and Quantum Gravity* **29**, 124016 (2012), [arXiv:1202.0839 \[gr-qc\]](#).
 - [135] J. R. Gair, M. Vallisneri, S. L. Larson, and J. G. Baker, *Living Reviews in Relativity* **16**, 7 (2013), [arXiv:1212.5575 \[gr-qc\]](#).
 - [136] A. Sesana, *Physical Review Letters* **116**, 231102 (2016), [arXiv:1602.06951 \[gr-qc\]](#).
 - [137] G. Hobbs, A. Archibald, Z. Arzoumanian, D. Backer, *et al.*, *Classical and Quantum Gravity* **27**, 084013 (2010), [arXiv:0911.5206 \[astro-ph.SR\]](#).
 - [138] M. A. McLaughlin, *Classical and Quantum Gravity* **30**, 224008 (2013), [arXiv:1310.0758 \[astro-ph.IM\]](#).
 - [139] Z. Arzoumanian, A. Brazier, S. Burke-Spolaor, S. J. Chamberlin, S. Chatterjee, B. Christy, J. M. Cordes, N. J. Cornish, K. Crowter, P. B. Demorest, X. Deng, T. Dolch, J. A. Ellis, R. D. Ferdman, E. Fonseca, N. Garver-Daniels, M. E. Gonzalez, F. Jenet, G. Jones, M. L. Jones, V. M. Kaspi, M. Koop, M. T. Lam, T. J. W. Lazio, L. Levin, A. N. Lommen, D. R. Lorimer, J. Luo, R. S. Lynch, D. R. Madison, M. A. McLaughlin, S. T. McWilliams, C. M. F. Mingarelli, D. J. Nice, N. Palliyaguru, T. T. Pennucci, S. M. Ransom, L. Sampson, S. A. Sanidas, A. Sesana, X. Siemens, J. Simon, I. H. Stairs, D. R. Stinebring, K. Stovall, J. Swiggum, S. R. Taylor, M. Vallisneri, R. van Haasteren, Y. Wang, W. W. Zhu, and NANOGrav Collaboration, *Astrophys. J.* **821**, 13 (2016), [arXiv:1508.03024](#).
 - [140] X. Siemens, J. Ellis, F. Jenet, and J. D. Romano, *Classical and Quantum Gravity* **30**, 224015 (2013), [arXiv:1305.3196 \[astro-ph.IM\]](#).
 - [141] A. Sesana, *Astrophys. J.* **719**, 851 (2010), [arXiv:1006.0730 \[astro-ph.CO\]](#).
 - [142] A. Sesana, *MNRAS* **433**, L1 (2013), [arXiv:1211.5375 \[astro-ph.CO\]](#).
 - [143] A. Sesana and A. Vecchio, *Phys. Rev. D* **81**, 104008 (2010), [arXiv:1003.0677 \[astro-ph.CO\]](#).
 - [144] A. Sesana and A. Vecchio, *Classical and Quantum Gravity* **27**, 084016 (2010), [arXiv:1001.3161 \[astro-ph.CO\]](#).
 - [145] E. A. Huerta, S. T. McWilliams, J. R. Gair, and S. R. Taylor, *Phys. Rev. D* **92**, 063010 (2015), [arXiv:1504.00928 \[gr-qc\]](#).
 - [146] M. Enoki and M. Nagashima, *Progress of Theoretical Physics* **117**, 241 (2007), [astro-ph/0609377](#).
 - [147] V. Ravi, J. S. B. Wyithe, R. M. Shannon, G. Hobbs, and R. N. Manchester, *MNRAS* **442**, 56 (2014), [arXiv:1404.5183](#).
 - [148] S. R. Taylor, J. Simon, and L. Sampson, (2016), [arXiv:1612.02817 \[astro-ph\]](#).
 - [149] S. R. Taylor, E. A. Huerta, J. R. Gair, and S. T. McWilliams, *Astrophys. J.* **817**, 70 (2016), [arXiv:1505.06208 \[gr-qc\]](#).
 - [150] S. Hochreiter and J. Schmidhuber, *Neural Comput.* **9**, 1735 (1997).